



**TUGAS AKHIR - TM 141585**

**PENDETEKSIAN DAN PELACAKAN 2D MULTI OBJEK  
SECARA REAL-TIME DENGAN MENGGUNAKAN  
KAMERA TUNGAL**

**MUHAMAD AL IMRON  
NRP. 2109 100 142**

**Dosen Pembimbing  
Arif Wahyudi ST., MT., Ph.D.  
NIP. 19730322 200112 1 001**

**JURUSAN TEKNIK MESIN  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
SURABAYA  
2015**

*halaman ini sengaja dikosongkan*



**FINAL PROJECT - 141585**

**DETECTION AND TRACKING OF MULTIPLE 2D  
OBJECTS IN REALTIME USING  
A SINGLE CAMERA**

**MUHAMAD AL IMRON  
NRP. 2109 100 142**

**Advisor  
Arif Wahyudi ST., MT., Ph.D.  
NIP. 19730322 200112 1 001**

**DEPARTMENT OF MECHANICAL ENGINEERING  
FACULTY OF INDUSTRIAL TECHNOLOGY  
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY  
SURABAYA  
2015**

*halaman ini sengaja dikosongkan*

## LEMBAR PENGESAHAN

### **PENDETEKSIAN DAN PELACAKAN 2D MULTI OBJEK SECARA *REAL-TIME* DENGAN MENGGUNAKAN KAMERA TUNGGAL**

#### **TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat Memperoleh Gelar  
Sarjana Teknik Mesin pada Bidang Studi Manufaktur  
Program Studi S-1  
Jurusan Teknik Mesin  
Fakultas Teknologi Industri  
Institut Teknologi Sepuluh Nopember Surabaya

Oleh :

**MUHAMAD AL IMRON**

NRP. 2109 100 142

Disetujui oleh Tim Penguji Tugas Akhir

1. Arif Wahyudi, ST., MT., Ph.D.  
(NIP.19730322 200112 1 001) .....  
(Pembimbing)
2. Ir. Bambang Pramujati, M.Sc.Eng., Ph.D.  
(NIP.19691203 199403 1 001) .....  
(Penguji 1)
3. Dr.Eng. Unggul Wasiwitono, ST., M.Eng.Sc.  
(NIP.19780510 200112 1 001) .....  
(Penguji 2)

**SURABAYA**

**JULI, 2015**

## **PENDETEKSIAN DAN PELACAKAN 2D MULTI OBJEK SECARA REAL-TIME DENGAN MENGGUNAKAN KAMERA TUNGGAL**

**Nama Mahasiswa** : Muhamad Al Imron  
**NRP** : 2109 100 142  
**Jurusan** : Teknik Mesin FTI-ITS  
**Dosen Pembimbing** : Arif Wahyudi, ST. MT. Ph.D.

### **Abstrak**

*Dewasa ini teknologi transportasi dunia berkembang sangat pesat. Salah satu dari teknologi tersebut adalah mendeteksi dan melacak mobil di sekitar deretan beberapa mobil. Hal tersebut sangat membantu pengemudi melakukan tindakan tertentu atau bemanuver untuk menghindari kejadian yang tidak diinginkan seperti kecelakaan dan bersenggolan. Salah satu metode pelacakan adalah dengan menggunakan tracking-by-detection, dimana pelacakan dilakukan pada semua frame yang didapatkan. Tahapan yang dilakukan dalam proses pendeteksian dan pelacakan antara lain tahap preprocessing, pendeteksian objek, dan pelacakan objek. Pendeteksian dan pelacakan dilakukan berdasarkan bidang kartesian xy yang akan dianalisa pada setiap frame berdasarkan bentuk dan histogram warna Red Green Blue (RGB) pada objek.*

*Dalam penelitian awal ini, objek berbentuk lingkaran digunakan sebagai penyederhanaan plat nomor yang akan dideteksi dan dilacak menggunakan kamera tunggal secara real-time. Multi objek digunakan sebagai pengganti jika kemungkinan mobil yang melintas lebih dari satu. Hasil pendeteksiaan tersebut akan diolah dan dianalisa apakah program tersebut bisa berjalan sesuai dengan yang diharapkan yaitu tanpa tertukar meski objek berpapasan.*

*Hasil yang didapatkan dari pendeteksian secara real-time tidak ada yang mencapai kemampuan pendeteksian 100%. Dalam*



*keadaan diam, kemampuan pendeteksian yang diperoleh berkisar diatas 95%. Objek terjauh yang bisa dideteksi secara real-time yaitu objek dengan radius 20 piksel. Tidak terjadi kesalahan pendeteksian pada saat objek berpapasan.*

***Kata kunci*** : mendeteksi, melacak, detection-by-tracking, frame, warna Red Green Blue(RGB), real-time, multi objek

## DETECTION AND TRACKING OF MULTIPLE 2D OBJECTS IN REALTIME USING A SINGLE CAMERA

**Name** : Muhamad Al Imron  
**NRP** : 2109 100 142  
**Major** : Mechanical Engineering FTI-ITS  
**Advisor** : Arif Wahyudi, ST. MT. Ph.D.

### ***Abstract***

*Nowadays transportation technology is developing very rapidly. One of these technologies is detection and tracking of a car around several cars in a street. This technology helps a driver perform a certain action to avoid unwanted incidents such as accidents and touched. One of the tracking method is using tracking-by-detection. This tracking method was done by detecting all obtained frames in sequence frames . The steps of this process are the preparation stage, preprocessing, object detection, and object tracking. The detection and tracking is based on Cartesian xy, field will be analyzed in each frame based on shape and color Red Green Blue (RGB) histogram on the object.*

*In this research, a circle-shaped object was used as a simplification of the number of object that will be detected and tracked using a single camera in real-time. Multi object was used as a substitute if the possibility of a passing object is more than one. The detection results will be processed and analyzed whether such programming are able to run as expected.*

*The results obtained from the detection in real-time have reached 100% detection capability. In the quiescent state, the detection capability reach 95%. Most distant object that can be detected in real-time that is an object with a radius of 20 pixels.*

**Keywords:** *detecting, tracking, detection-by-tracking, frame, color Red Green Blue (RGB), real-time, multi-object*





## KATA PENGANTAR

**Bismillahirrohmanirohim.**

Alhamdulillahil'alamini, segala puji bagi Allah SWT, atas segala rahmat dan karunia-Nya yang tak terhingga sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul:

### **“Pendeteksian dan Pelacakan 2D Multi Objek Secara Real-Time Dengan Menggunakan Kamera Tunggal”**

Terselesaikannya Tugas Akhir ini tidak terlepas dari bantuan banyak pihak. Oleh karena itu melalui lembar ini penulis ingin mengucapkan terima kasih dan penghormatan sebesar-besarnya kepada pihak-pihak sebagai berikut.

1. Allah SWT, karena limpahan rahmat dan karuniaNya lah penulis dapat menyelesaikan Tugas Akhir dan juga perkuliahan di Teknik Mesin FTI-ITS.
2. Ibu dan ayah penulis, Komsiyah dan Djaenuri yang tiada henti memberikan dukungan penuh kepada penulis selama ini sehingga dapat menyelesaikan Tugas Akhir ini
3. Saudara penulis, Siti Nur Asiyah dan Wijayanti Nurul Khotimah yang telah memberikan dukungan kepada penulis untuk terus semangat dan membantu untuk menyelesaikan Tugas Akhir ini.
4. Bapak Ir. Bambang Pramujati, M.Sc.Eng., Ph.D. selaku Ketua Jurusan Teknik Mesin FTI-ITS.

5. Bapak Arif Wahyudi, ST., MT., Ph.D. selaku dosen pembimbing dari penulis yang telah memberikan banyak arahan dan nasihat kepada penulis dalam menyelesaikan Tugas Akhir ini.
6. Bapak Dr. Eng. Harus Laksa Guntur, ST., M.Eng. selaku dosen wali dari penulis yang selalu memberi nasihat kepada penulis selama menjalani perkuliahan di Teknik Mesin FTI-ITS.
7. Teman-teman super sepermainan dari penulis, Hendra, Nanang, Dharmo, Deni, Ranga, Uza, Solikhin, dan Veri yang selalu memberikan semangat, dukungan, dan menemani keseharian penulis.
8. Teman-teman M-52 dan Lab. P3 yang telah ramah dan baik hati membantu penulis selama berada di Mesin FTI-ITS.
9. Pihak-pihak yang tidak dapat penulis sebutkan satu per satu yang telah membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis sadar bahwa penulisan tugas akhir ini memiliki banyak kekurangan, untuk itu kritik dan saran yang membangun sangat diharapkan demi perbaikan dan kesempurnaan tugas akhir ini. Semoga tulisan ini dapat bermanfaat bagi semua. Amin.

Surabaya,

Agustus 2015

Penulis

## DAFTAR ISI

LEMBAR PENGESAHAN .....	v
Abstrak .....	vii
Abstrack .....	ix
KATA PENGANTAR .....	xi
DAFTAR ISI .....	xiii
DAFTAR GAMBAR .....	xvii
DAFTAR TABEL .....	xxi
<b>BAB I PENDAHULUAN .....</b>	<b>1</b>
1.1 Latar Belakang .....	2
1.2 Perumusan Masalah .....	3
1.3 Batasan Masalah .....	3
1.4 Tujuan Penelitian .....	3
1.5 Manfaat Penelitian .....	4
1.6 Sistematika Penulisan .....	5
<b>BAB II KAJIAN PUSTAKA .....</b>	<b>7</b>
2.1 Pendeteksian dan Pelacakan Objek .....	7
2.2 Sistem Visual Manusia.....	8
2.3 Pembentukan Citra .....	11
2.4 Model Citra .....	12
2.5 Digitalisasi Citra .....	14
2.6 Digitalisasi Spasial .....	16
2.7 Peningkatan Mutu Citra .....	18
2.8 Kontur dan Representasi .....	22
2.9 Transformasi Hough Lingkaran .....	24
2.10.1 Jari-jari Lingkaran Diketahui .....	25
2.10.2 Jari-jari Lingkaran tidak Diketahui .....	26
2.10 Evaluasi Kemampuan Pendeteksian .....	27
2.11 OpenCV .....	28



<b>BAB III METODE PENELITIAN .....</b>	<b>31</b>
3.1 Diagram Alir Penelitian .....	31
3.2 Kajian Pustaka .....	32
3.3 Identifikasi Masala .....	33
3.4 Perumusan Masalah .....	33
3.5 Perancangan dan Pengembangan Program .....	33
3.5.1 Variabel Range Warna Threshold .....	33
3.5.2 Pencarian Variabel Pendeteksian Objek .....	34
3.6 Pembahasan.....	37
3.7 Kesimpulan dan Saran .....	37
<b>BAB IV PERCOBAAN .....</b>	<b>39</b>
4.1 Program Pendeteksi Objek.....	40
4.2 Pencarian Nilai Variabel Pendeteksian .....	45
4.2.1 Variabel Range Warna Threshold .....	45
4.2.2 Pencarian Variabel Pendeteksian Objek.....	47
<b>BAB V PEMBAHASAN.....</b>	<b>53</b>
5.1 Pencarian Range Variabel Pendeteksian .....	53
5.2 Objek Tunggal .....	55
5.2.1 Kemampuan Pendeteksian Objek Diam.....	55
5.2.2 Pendeteksian Objek Merah .....	58
5.2.3 Pendeteksian Objek Hijau .....	65
5.2.4 Pendeteksian Objek Ungu .....	71
5.2.5 Perbandingan Hasil Pendeteksian <i>Real-Time</i> .....	76
5.3 Multi Objek.....	78
5.3.1 Pendeteksian Multi Objek Diam .....	78
5.2.2 Pendeteksian Multi Objek Bergerak ...	82
5.4 Pendeteksian Jarak Terjauh Objek Terdeteksi .	84



<b>BAB VI PENUTUP .....</b>	<b>87</b>
6.1   Kesimpulan .....	87
6.2   Saran .....	88

**DAFTAR PUSTAKA**  
**LAMPIRAN**  
**BODATA PENULIS**



## DAFTAR GAMBAR

Gambar 2.1	Sistem Visual Manusia .....	8
Gambar 2.2	Ilustrasi Penglihatan pada Mata Manusia Saat Melihat Pohon Palm .....	9
Gambar 2.3	Garis Seperti Lipatan pada Daerah Batas Tingkat Intensitas .....	10
Gambar 2.4	Tampilan Kotak di Tengan Mempunyai Intensitas yang Sama Tetapi Berkesan Lebih Gelap Apabila Background Lebih Terang .....	10
Gambar 2.5	Pembentukan Gambar 2-D.....	11
Gambar 2.6	Prinsip Sensor Bersusun .....	12
Gambar 2.7	Koordinat Titik di dalam Citra .....	13
Gambar 2.8	Pembentukan Citra .....	14
Gambar 2.9	<i>Sampling</i> .....	16
Gambar 2.10	Hubungan Antara Elemen Gambar dan Elemen Matrik .....	16
Gambar 2.11	Perbedaan Tampilan ketika Citra Mengalami Pengurangan Pikel .....	17
Gambar 2.12	<i>Contrast-Stretching Function</i> .....	19
Gambar 2.13	Transformasi <i>Threshold</i> .....	20
Gambar 2.14	Empat Dasar Tipe Citra dan Histogramnya ...	21
Gambar 2.15	Kontur .....	23
Gambar 2.16	Kode Rantai .....	23
Gambar 2.17	Proses Penumpukan <i>Voting</i> untuk Mendeteksi Lingkaran .....	25
Gambar 2.18	Hasil Transformasi Hough.....	27
Gambar 3.1	Diagram Alir Penelitian .....	31
Gambar 3.2	Objek.....	33
Gambar 3.3	Kamera Usb The Imaging Source DFK31BU03 .....	34
Gambar 3.4	Segmentasi Objek .....	35
Gambar 3.5	Pengantungan Objek dengan Seutas Tali .....	37

Gambar 4.1	Diagram Alir Proses Perancangan Program ...	39
Gambar 4.2	Diagram Alir Program Pendeteksian .....	40
Gambar 4.3	Citra Asli .....	42
Gambar 4.4	Diagram Alir Proses <i>Preprocessing</i> .....	42
Gambar 4.5	Citra <i>Threshold</i> .....	43
Gambar 4.6	Citra <i>Threshold</i> Hasil <i>Erode</i> 2x2 piksel .....	43
Gambar 4.7	Citra <i>Threshold</i> Hasil <i>dilade</i> 9x9 Piksel .....	44
Gambar 4.8	Hasil <i>Smoothing</i> Citra <i>Treshold</i> .....	44
Gambar 4.9	Diagram Alir Pencarian <i>Range</i> Warna .....	45
Gambar 4.10	<i>Trackbar</i> Pencari Perubahan Citra RGB ke Citra <i>Threshold</i> .....	46
Gambar 4.11	Citra <i>Threshold</i> dari <i>Trackbar</i> .....	46
Gambar 4.12	Informasi Hasil Pencarian <i>Range</i> dengan <i>trackbar</i> .....	47
Gambar 4.13	Diagram Alir Pencarian Variabel .....	48
Gambar 4.14	Objek Uji Coba 1 .....	49
Gambar 4.15	Hubungan Inverse Rasio Resolusi, Parameter 1, dan Parameter 2 terhadap Proses Pendeteksian .....	50
Gambar 4.16	Tipe Kesalahan Pendeteksian Objek .....	49
Gambar 5.1	Objek Asli dan Hasil <i>Threshold</i> .....	53
Gambar 5.2	Hasil Pendeteksian Objek Diam Kondisi 1 .....	56
Gambar 5.3	Perbandingan Hasil <i>Preprocessing</i> Objek Merah Terdeteksi dan Objek Merah Tidak Terdeteksi .....	57
Gambar 5.4	Evaluasi Pendeteksian .....	57
Gambar 5.5	Objek $\theta_o = 25^\circ$ dengan $\theta_f = 1^\circ$ .....	61
Gambar 5.6	Hasil Pendeteksian Objek Bergerak $\theta_o$ Sebesar $60^\circ$ .....	61
Gambar 5.7	Lintasan Pertama Objek Merah dengan $\theta_o = 45^\circ$ .....	64
Gambar 5.8	Lintasan Pertama Objek Merah dengan $\theta_o = 60^\circ$ .....	64



Gambar 5.9	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $20^\circ$ .....	67
Gambar 5.10	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $60^\circ$ .....	67
Gambar 5.11	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $25^\circ$ .....	70
Gambar 5.12	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $55^\circ$ .....	70
Gambar 5.13	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $20^\circ$ .....	73
Gambar 5.14	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $55^\circ$ .....	73
Gambar 5.15	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $13^\circ$ .....	75
Gambar 5.16	Hasil Pendeteksian Objek Bergerak $\theta_0$ Sebesar $60^\circ$ .....	76
Gambar 5.17	Citra Hasil Pendeteksian dan Evaluasi <i>Frame</i> ke-1 dan ke-4 .....	79
Gambar 5.18	Citra <i>Threshold</i> Hasil Evaluasi <i>Frame</i> ke-1 dan ke-4 .....	79
Gambar 5.19	Gambar 5.19. Citra <i>Threshold</i> Hasil Evaluasi <i>Frame</i> ke-46 .....	32
Gambar 5.20	Citra Hasil Pendeteksian dan Evaluasi <i>Frame</i> ke-8 dan ke-94 .....	81
Gambar 5.21	Citra <i>Threshold</i> Hasil Evaluasi <i>Frame</i> ke-8 dan ke-94 .....	82
Gambar 5.22	Pendeteksian Multi Objek Bergerak .....	83
Gambar 5.23	Citra <i>Threshold</i> Multi Objek .....	83
Gambar 5.14	Perbedaan Hasil Pendeteksian <i>Real-time</i> dan Evaluasi .....	84
Gambar 5.15	Citra Pendeteksian .....	85
Gambar Lampiran 2.	Ruangan Uji Pencarian Jarak Terjauh Pendeteksian	
Gambar Lampiran 2b.	Ruangan Uji Pendeteksian Objek	





## DAFTAR TABEL

Tabel 2.1	Skala Keabuan dan Rentang Nilai .....	23
Tabel 5.1	Hasil Pencarian Range RGB .....	53
Tabel 5.2	Jarak Warna RGB Objek .....	54
Tabel 5.3	<i>Range</i> Bagian Atas dan Bagian Bawah .....	54
Tabel 5.4	Hasil Pendeteksian Objek Diam Kondisi 1.....	56
Tabel 5.5	Hasil Pendeteksian Objek Diam Kondisi 2.....	58
Tabel 5.6	Hasil Pendeteksian Objek Merah pada Kondisi 1 .....	59
Tabel 5.7	Hasil Evaluasi Objek Merah Tidak Terdeteksi	60
Tabel 5.8	Hasil Pendeteksian Objek Hijau pada Kondisi 2 .....	62
Tabel 5.9	Hasil Evaluasi Objek Merah Tidak Terdeteksi Kondisi 2 .....	63
Tabel 5.10	Hasil Pendeteksian Objek Hijau pada Kondisi 1 .....	65
Tabel 5.11	Hasil Evaluasi Objek Hijau Tidak Terdeteksi Kondisi 1 .....	66
Tabel 5.12	Hasil Pendeteksian Objek Hijau pada Kondisi 2 .....	68
Tabel 5.13	Hasil Evaluasi Objek Hijau Tidak Terdeteksi Kondisi 2 .....	69
Tabel 5.14	Hasil Pendeteksian Objek Ungu pada Kondisi 1 .....	71
Tabel 5.15	Hasil Evaluasi Objek Ungu Tidak Terdeteksi Kondisi 1 .....	72
Tabel 5.16	Hasil Pendeteksian Objek Ungu pada Kondisi 2 .....	74
Tabel 5.17	Hasil Pendeteksian Objek Ungu pada Kondisi 2 .....	74
Tabel 5.18	Perbandingan Kemampuan Pendeteksian <i>Real- Time</i> . ....	77

Tabel 5.19	Hasil Evaluasi Pendeteksian <i>Real-Time</i> .....	77
Tabel 5.20	Hasil Pendeteksian Multi Objek tidak Bergerak Kondisi 1 .....	78
Tabel 5.21	Hasil Evaluasi Objek Tidak Terdeteksi.....	80
Tabel 5.22	Hasil Pendeteksian Multi Objek tidak Bergerak Kondisi 2 .....	81
Tabel 5.23	Hasil <i>Real-time</i> Pendeteksian Multi Objek Bergerak.....	82
Tabel Lampiran 4a.	Pendeteksian Real-Time Objek Merah Kondisi 1	
Tabel Lampiran 4b.	Pendeteksian Real-Time Objek Merah Kondisi 2	
Tabel Lampiran 4c.	Pendeteksian Real-Time Objek Hijau Kondisi 1	
Tabel Lampiran 4d.	Pendeteksian Real-Time Objek Hijau Kondisi 2	
Tabel Lampiran 4e.	Pendeteksian Real-Time Objek Ungu Kondisi 1	
Tabel Lampiran 4f.	Pendeteksian Real-Time Objek Ungu Kondisi 2	

# BAB I PENDAHULUAN

## 1.1. Latar Belakang

Dewasa ini teknologi transportasi dunia sangat meningkat. Salah satu dari teknologi tersebut adalah mendeteksi keberadaan objek-objek di sekitar mobil. Bagaimana mobil bisa mendeteksi keberadaan objek yang kemudian dilakukan tindakan agar mobil tetap aman dan nyaman. Untuk menjaga keamanan mobil, perlu dianalisa jarak aman mobil tersebut dengan mobil disekitarnya supaya tidak terjadi hal yang tidak diinginkan seperti kecelakaan. Objek utama yang dideteksi adalah mobil di sekitarnya dengan meninjau keberadaan plat nomor kendaraan.

Pengolahan citra (*image processing*) adalah teknik mengolah citra yang mentransformasikan citra masukan menjadi citra lain agar keluaran memiliki kualitas yang lebih baik dibandingkan kualitas citra masukan. Pengolahan citra sangat bermanfaat, diantaranya adalah untuk meningkatkan kualitas citra, menghilangkan cacat pada citra, mengidentifikasi objek, dan untuk menggabungkan dengan bagian citra yang lain. Dengan memanfaatkan teknologi tersebut, maka diharapkan adanya suatu aplikasi yang dapat menangkap suatu objek yang ada di depan kamera, bisa mengidentifikasi jenis objek serta melakukan *tracking* objek secara *real-time*.

Dalam prakteknya, ada dua pendekatan untuk melacak gerakan suatu objek. Pendekatan pertama disebut pendekatan *tracking-by-detection*. Pada pendekatan ini, untuk melacak suatu objek dilakukan pendeteksian objek tersebut pada setiap *frame* gambar bergerak atau video yang diamati, guna menentukan posisi objek itu pada setiap *frame*-nya. Pendekatan kedua disebut pendekatan *detection-by-tracking*. Pada pendekatan ini, pendeteksian objek hanya dilakukan pada satu atau beberapa *frame*



pertama saja, lalu untuk mengetahui gerak atau posisi objek pada *frame-frame* selanjutnya cukup melakukan penjejakan perubahan gerak antara *frame* saat ini terhadap *frame-frame* sebelumnya. Jadi kita tidak perlu menjalankan algoritma pendeteksian objek disetiap *frame*-nya.

Pendekatan *detection-by-tracking* secara komputasi lebih efisien dibandingkan pendekatan *tracking-by-detection* apalagi pada setiap *frame*-nya objek yang menjadi target ternyata hanya melakukan perubahan kecil saja. Untuk mengetahui posisi (bahkan dalam aplikasi tertentu kita juga perlu menentukan orientasi, kecepatan, dan percepatan) suatu objek, pendeteksian dan pelacakan saling terkait erat satu sama lain karena pendeteksian diperlukan pada fase awal sedangkan pelacakan digunakan pada fase selanjutnya[1].

Dalam tugas akhir ini, direncanakan objek yang diidentifikasi berbentuk bulat sebagai pengganti keberadaan plat nomor kendaraan dengan menggunakan kamera tunggal secara *real-time* berdasarkan histogram warna RGB. Multi objek digunakan sebagai pengganti jika kemungkinan mobil yang melintas lebih dari satu. Hasil dari pendeteksiaan tersebut kemudian diolah dan dianalisa apakah program tersebut bisa berjalan sesuai yang diharapkan.

## **1.2. Perumusan Masalah**

Berdasarkan latar belakang tersebut diatas maka dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana mendeteksi sebuah objek secara *real-time*.
2. Bagaimana mendeteksi objek yang bergerak secara *real-time*.
3. Bagaimana menandai multi objek tanpa tertukar.



### 1.3. Batasan Masalah

Agar tujuan dari penulisan tugas akhir ini dapat tercapai maka diperlukan adanya batasan masalah sebagai berikut:

1. Deteksi objek dengan menggunakan jenis kamera Usb The Imaging Source DFK 23UP031.
2. Target yang akan dideteksi berupa objek berbentuk bulat.
3. Pendeteksian dan pelacakan dilakukan di dalam ruangan.
4. Bahasa pemrograman yang digunakan adalah C++.
5. Menggunakan *library* openCV 3.0.0.beta.
6. Menggunakan perangkat lunak lengkap (*suite*) microsoft visual studio 2012.
7. Pendeteksian hanya digunakan untuk mengetahui posisi objek dalam 2D atau dalam bidang kartesian xy.

### 1.4. Perumusan Masalah

Tujuan dari penelitian ini adalah sebagai berikut:

1. Mendeteksi sebuah objek berbentuk bulat secara *real-time*.
2. Melacak objek bergerak secara *real-time*.
3. Menandai multi objek tanpa tertukar.

### 1.5. Manfaat

Manfaat dari hasil pembuatan tugas akhir ini antara lain

1. Untuk mengetahui tingkat kemampuan pendeteksian, efisiensi serta pemilihan parameter yang baik pada penerapan metode *detection-by-tracking* untuk pelacakan suatu objek.
2. Untuk membuat sistem yang bisa mengetahui posisi objek dalam 2D atau bidang kartesian xy sehingga akan mempermudah mengetahui posisi objek sebenarnya dalam 3D.

3. Sebagai dasar yang akan diimplementasikan untuk mengukur jarak objek tersebut terhadap kamera yang digunakan.

## **1.6. Sistematika Penulisan**

Buku tugas akhir ini terdiri dari beberapa Bab, yang dijelaskan sebagai berikut.

### **BAB I PENDAHULUAN**

Bab ini berisi penjelasan mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan.

### **BAB II TINJAUAN PUSTAKA**

Bab ini berisi penjelasan secara detail mengenai dasar-dasar ilmu yang menjadi referensi pendukung dalam mengerjakan tugas akhir.

### **BAB III METODOLOGI PENELITIAN**

Bab ini berisi penjelasan mengenai metode penelitian yang digunakan dalam mengerjakan tugas akhir.

### **BAB IV PERCOBAAN**

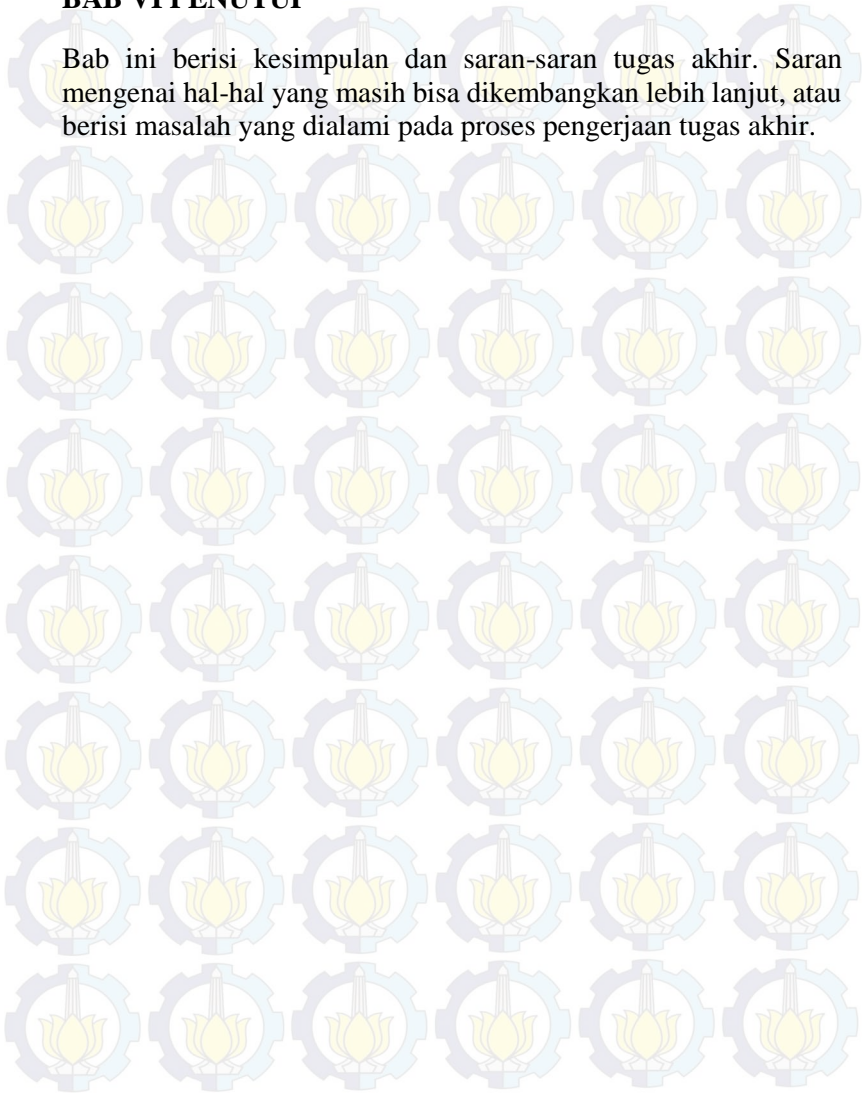
Bab ini berisi mengenai desain dari algoritma pendeteksian dan pelacakan dari tugas akhir.

### **BAB V PEMBAHASAN**

Bab ini berisi penjelasan mengenai data hasil pengujian dan pembahasannya.

## **BAB VI PENUTUP**

Bab ini berisi kesimpulan dan saran-saran tugas akhir. Saran mengenai hal-hal yang masih bisa dikembangkan lebih lanjut, atau berisi masalah yang dialami pada proses pengerjaan tugas akhir.







## BAB II KAJIAN PUSTAKA

Tugas akhir dengan judul “Pendeteksian Multi Objek 2D Menggunakan Kamera Tunggal Secara *Real-Time*” ini melibatkan beberapa teori penunjang. Bab ini membahas tentang pendeteksian dan pelacakan objek, teori mengenai pengolahan citra, dan teori mengenai openCV.

### 2.1. Pendeteksi dan Pelacakan Objek

Pendeteksian dan pelacakan objek sudah menjadi perhatian peneliti dewasa ini. Mulai dari pendeteksian objek tunggal sampai dengan multi objek sudah diusulkan pada *paper-paper* beberapa tahun terakhir. Sebuah objek tunggal telah diusulkan dengan (RS-PFRSR) *robust combination of particle filter and sparse representation* [2]. Multi objek telah diusulkan dengan metode *hieraarchical particle filter* [3].

Metode RS-PFRSR dimaksudkan untuk mengecilkan kemungkinan adanya gerakan yang menyimpang pada saat pendeteksian. Metode ini dilakukan pada MATLAB dan digunakan sebagai objek eksperimen seperti wajah manusia, mobil, dan batu. Pendeteksian dan pelacakan menggunakan 600 *frame* sebagai data latih dan 15 *template set* sebagai *template*. Eksperimen pendeteksian pada objek tersebut dilakukan dengan pendekatan *template* yang disediakan terhadap data latih.

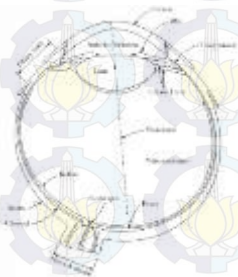
Metode multi objek dengan *hieraarchical particle filter* menggunakan acuan *range* warna yang terdapat pada objek disetiap *frame*-nya. Pelacakan menggunakan 1000 *frame* data latih dengan kecepatan 50 *fps* (*frame per second*). Eksperimen pendeteksian dan pelacakan menggunakan objek dengan gambar manusia dengan pendekatan histogram warna *red gren blue* (RGB) dari setiap *frame*-nya.



Kedua metode di atas menggunakan *particle filter* yaitu melacak objek dengan melakukan pendekatan objek yang berganti untuk setiap waktu. Untuk metode RS-PFRSR berdasarkan *template set* sedangkan *hierarchical particle filter* berdasarkan histogram warna di setiap waktu. Pada saat mendeteksi dan melacak memungkinkan adanya *error* (kesalahan) yang terjadi. *Error* yang terjadi pada pendeteksian dan pelacakan bisa dianalisa pada setiap *frame* yang digunakan. *Error* tersebut dicari dengan membandingkan hasil *frame* yang tidak berhasil mendeteksi keberadaan objek dengan *frame* yang berhasil mendeteksi objek dimulai dari *frame* yang terdapat objeknya.

## 2.2. Sistem Visual Manusia

Konsep sistem visual manusia dalam pengolahan citra sangat identik dengan proses pengolahan citra. Ketika manusia menangkap citra di mata, maka prinsip tersebut juga diterapkan oleh pengolahan citra untuk mengambil citra dari suatu alat sensor citra. Dari informasi citra yang ditangkap oleh mata, kemudian diproses untuk mengidentifikasi citra tersebut.



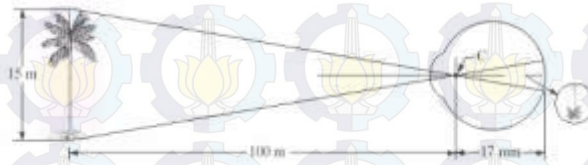
Gambar 2.1. Sistem Visual Manusia [4]

Gambar 2.1 menunjukkan system penangkapan citra pada mata manusia. Intensitas cahaya ditangkap oleh iris dan diteruskan ke bagian retina mata. Kemudian bayangan objek mata dibentuk

pada retina dengan mengikuti konsep sistem optik dimana fokus lensa terletak antara retina dan lensa mata. Mata dan syaraf otak dapat menginterpretasi bayangan yang merupakan objek pada posisi terbalik.

Pada sistem mata manusia terdapat bagian fovea. Fovea di bagian retina terdiri dari dua jenis reseptor yaitu *cone receptor* yang sensitif terhadap warna dan *rod receptor* yang sensitif terhadap cahaya. Selain itu pada retina mata terdapat bagian yang tidak mengandung *receptor* sehingga tidak dapat menerima dan menginterpretasi informasi yang biasa disebut *blind spot*.

Lensa pada sebuah kamera mempunyai *focal length* yang tetap dan fokus pada variasi jarak tertentu seperti halnya pada mata manusia. Ilustrasi dari formasi gambar yang ditangkap manusia terlihat seperti Gambar 2.2.



Gambar 2.2. Ilustrasi Penglihatan pada Mata Manusia Saat Melihat Pohon Palm[4]

Pada Gambar 2.2 menunjukkan dimensi sebuah gambar yang terbentuk pada retina dengan titik C adalah titik pusat dari lensa. Pada ilustrasi tersebut terlihat bahwa pohon palm yang mempunyai tinggi 15 m dengan jarak dari titik fokus atau penglihatan 100 m dengan *focal length* 17 mm. Intensitas dari masing-masing daerah pada Gambar 2.3 adalah konstan tetapi pada kenyataannya *pattern brightness* sangat kuat sehingga membentuk seperti garis lipatan di dekat batas masing-masing daerah tersebut.

Gambar 2.4 menyatakan suatu model *contrast simultan*, dimana kemunculan kotak kecil ditengah berkesan memiliki intensitas yang semakin lebih gelap apabila warna *background* menjadi lebih terang, padahal kenyataannya intensitas kotak kecil tersebut sama. Hal ini menyatakan bahwa daerah *brightness* yang diterima tidak bergantung sederhana pada intensitasnya.



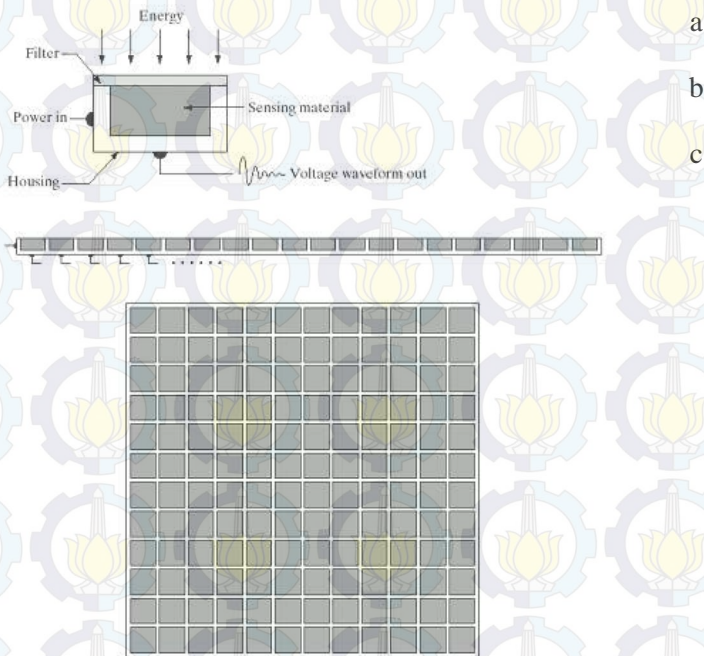
Gambar 2.3. Garis Seperti Lipatan pada Daerah Batas Tingkat Intensitas[4]



Gambar 2.4. Tampilan Kotak di Tengan Mempunyai Intensitas yang Sama Tetapi Berkesan Lebih Gelap Apabila Background Lebih Terang [5]

### 2.3. Pembentukan Citra

Banyak dari citra yang terjadi akibat kombinasi dari sebuah cahaya dan pantulan energi yang diserap dari elemen sebuah gambar.



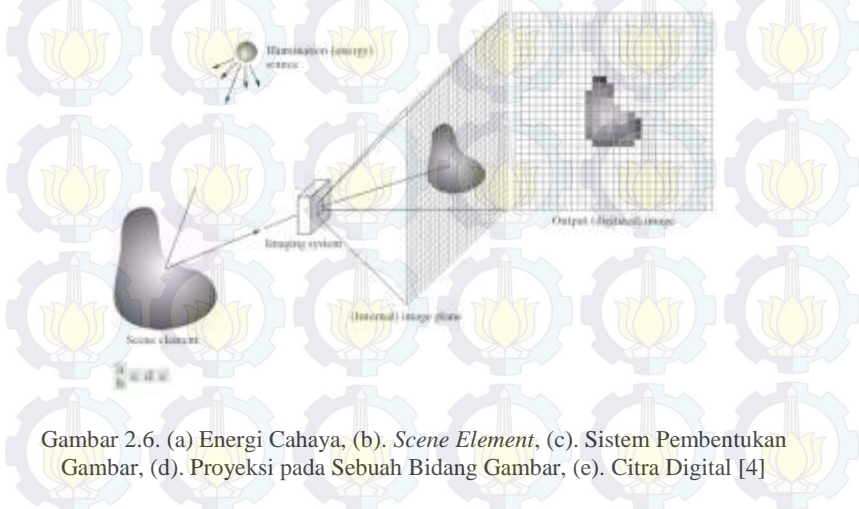
Gambar 2.5(a). *Single Imaging Sensor*, (b). *Line Sensor*, (c). *Array Sensor* [4]

Gambar 2.5 menunjukkan tiga prinsip sensor pembentuk yang mentransformasikan cahaya menjadi citra digital. Pada dasarnya sangat sederhana, jika ada energi yang datang kemudian dirubah



menjadi voltase yang dihasilkan dari kombinasi *input electrical power* dan material sensor yang sangat resposif terhadap partikel energi yang terdeteksi.

Gambar 2.6(c) menunjukkan pembentukan gambar bidang 2-D dari sebuah sensor. Beberapa *electromagnetic* dan perangkat sensor ultrasonik menggunakan format bersusun. Hal ini sangat sering ditemukan pada pengaturan kamera. Tipikal sensor dari kamera ini adalah *Charge Couple Device* (CCD) bersusun.



Gambar 2.6. (a) Energi Cahaya, (b). *Scene Element*, (c). Sistem Pembentukan Gambar, (d). Proyeksi pada Sebuah Bidang Gambar, (e). Citra Digital [4]

## 2.4. Model Citra

Citra dapat diartikan sebagai suatu fungsi intensitas cahaya dua dimensi, yang dinyatakan oleh  $f(x, y)$ , dimana nilai amplitudo dari  $f$  pada koordinat spasial  $(x, y)$  menyatakan intensitas (kecerahan) citra pada titik tersebut[4]. Gambar 2.7 memperlihatkan posisi koordinat pada bidang citra. Sistem koordinat yang diacu adalah sistem koordinat kartesian, dalam hal

ini sumbu mendatar menyatakan sumbu- $x$ , dan sumbu tegak menyatakan sumbu- $y$ .



Gambar 2.7. Koordinat Titik di dalam Citra

Karena cahaya merupakan bentuk energi, maka intensitas cahaya bernilai antara 0 sampai tidak berhingga,

$$0 \leq f(x, y) < \infty \dots\dots\dots (1)$$

Nilai  $f(x, y)$  sebenarnya adalah hasil kali dari [4]:

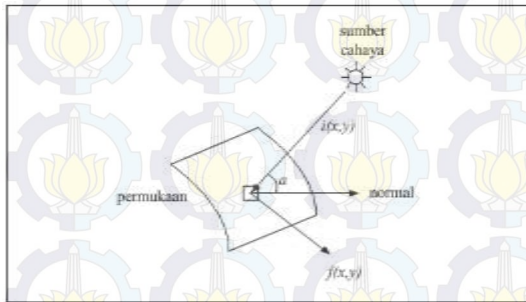
1.  $i(x, y)$  = jumlah cahaya yang berasal dari sumbernya (*illumination*), nilainya antara 0 sampai tidak berhingga, dan
2.  $r(x, y)$  = derajat kemampuan objek memantulkan cahaya (*reflection*), nilainya antara 0 dan 1.

Gambar 2.8 memperlihatkan proses pembentukan citra dimana sumber cahaya menyinari permukaan objek. Jumlah pancaran (*iluminasi*) cahaya yang diterima objek pada koodinat  $(x, y)$  adalah  $i(x, y)$ . Objek memantulkan cahaya yang diterimanya dengan derajat pantulan  $r(x, y)$ . Hasil kali antara  $i(x, y)$  dan  $r(x, y)$  menyatakan intensitas cahaya pada koordinat  $(x, y)$  yang ditangkap oleh sensor visual pada sistem optik. Jadi,

$$f(x, y) = i(x, y) \times r(x, y) \dots\dots\dots (2)$$

yang dalam hal ini,

$$0 \leq i(x, y) < \infty, 0 \leq r(x, y) \leq 1$$



Gambar 2.8. Pembentukan Citra [6]

Nilai  $i(x, y)$  ditentukan oleh sumber cahaya, sedangkan  $r(x, y)$  ditentukan oleh karakteristik objek di dalam gambar. Nilai  $r(x, y) = 0$  mengindikasikan penerapan total, sedangkan  $r(x, y) = 1$  menyatakan pemantulan total. Jika permukaan mempunyai derajat pemantulan nol, maka fungsi intensitas cahaya,  $f(x, y)$  juga nol. Sebaliknya, jika permukaan mempunyai derajat pemantulan 1, maka fungsi intensitas cahaya sama dengan iluminasi yang diterima oleh permukaan tersebut.

## 2.5. Digitalisasi Citra

Agar dapat diolah dengan komputer digital, maka suatu citra harus direpresentasikan secara numerik dengan nilai-nilai diskrit. Representasi citra dari fungsi malar menjadi nilai-nilai diskrit disebut digitalisasi. Citra yang dihasilkan inilah yang disebut citra digital (*digital image*). Pada umumnya citra digital

berbentuk empat persegi panjang dan dimensi ukurannya dinyatakan sebagai tinggi  $x$  lebar (atau lebar  $x$  panjang).

Citra digital yang tingginya  $N$ , lebarnya  $M$ , dan memiliki  $L$  derajat keabuan dapat dianggap sebagai fungsi [5]:

$$f(x, y) \begin{cases} 0 \leq x \leq M \\ 0 \leq y \leq N \\ 0 \leq f \leq L \end{cases} \dots\dots\dots (3)$$

Citra digital yang berukuran  $N \times M$  lazim dinyatakan dengan matriks yang berukuran  $N$  baris dan  $M$  kolom sebagai berikut:

$$f(x, y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M) \\ f(1,0) & f(1,1) & \dots & f(1,M) \\ \vdots & \vdots & \ddots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M) \end{bmatrix}$$

Indeks baris ( $i$ ) dan indeks kolom ( $j$ ) menyatakan suatu koordinat titik pada citra, sedangkan  $f(i, j)$  merupakan intensitas (derajat keabuan) pada titik ( $i, j$ ). Masing-masing elemen pada citra digital (berarti elemen matriks) disebut *image element*, *picture element* atau piksel. Jadi, citra yang berukuran  $N \times M$  mempunyai  $NM$  buah piksel. Proses digitalisasi citra ada dua macam:

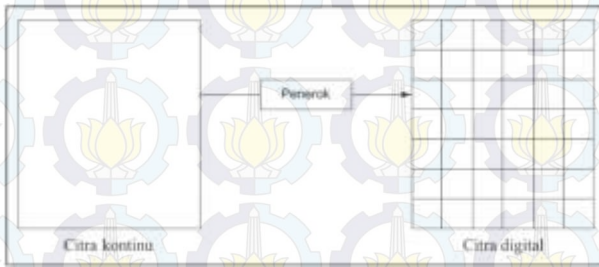
1. Digitalisasi spasial ( $x, y$ ), sering disebut sebagai penerokan (*sampling*).
2. Digitalisasi intensitas  $f(x, y)$ , sering disebut sebagai kuantisasi.

## 2.6. Digitalisasi Spasial

Secara umum digitalisasi spasial dilakukan dengan melakukan *sampling* pada *grid-grid* yang berbentuk bujur sangkar (kisi-kisi dalam arah horizontal dan vertikal) dari sebuah citra seperti terlihat pada Gambar 2.9. Digitalisasi spasial dapat

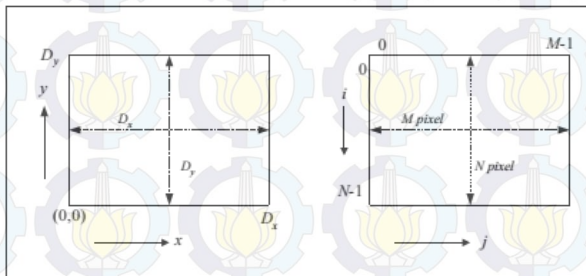


dikategorikan menjadi tiga [4]: (1) operasi *single-pixels*, (2) operasi *neighborhod*, dan (3) transformasi *geometric spacial*.



Gambar 2.9. *Sampling* [6]

Terdapat perbedaan antara koordinat gambar (yang di-*sampling*) dengan koordinat matriks (hasil digitalisasi). Titik asal (0, 0) pada gambar dan elemen (0,0) pada matriks tidak sama. Koordinat  $x$  dan  $y$  pada gambar dimulai dari sudut kiri bawah, sedangkan penomoran piksel pada matriks dimulai dari sudut kiri atas seperti terlihat pada Gambar 2.10.



Gambar 2.10. Hubungan Antara Elemen Gambar dan Elemen Matrik [6]

Dalam hal ini,

$$i = x \quad , \quad 0 \leq i \leq N - 1 \text{ dan}$$

$$j = (M - y), 0 \leq j \leq M - 1$$

Pembagian gambar menjadi ukuran tertentu menentukan resolusi (yaitu derajat rincian yang dapat dilihat) spasial yang diperoleh. Semakin tinggi resolusinya, yang berarti semakin kecil ukuran piksel (atau semakin banyak jumlah pikselnya), semakin halus gambar yang diperoleh karena informasi yang hilang akibat pengelompokan derajat keabuan pada penerokan semakin kecil. Gambar 2.11, terlihat perbedaan yang jelas pada gambar antara 256 x 256 piksel, 128 x 128 piksel, 64 x 64 piksel, dan 32 x 32 piksel.



Gambar 2.11. Perbedaan Tampilan ketika Citra Mengalami Pengurangan Piksel [5]

Langkah selanjutnya setelah proses penerokan (*sampling*) adalah kuantisasi. Proses kuantisasi membagi skala keabuan (0, L-1) menjadi  $L$  buah level yang dinyatakan dengan suatu harga bilangan bulat (integer), biasanya  $L$  diambil perpangkatan dari 2, seperti pada Tabel 2.1 dan persamaan 4.

$$L = 2^k \dots\dots\dots (4)$$

dalam hal ini,

$L$  = derajat keabuan dan  $k$  = bilangan bulat positif

Hitam dinyatakan dengan nilai derajat keabuan terendah, yaitu 0, sedangkan putih dinyatakan dengan nilai derajat keabuan tertinggi, misalnya 15 untuk 16 level. Jumlah bit yang dibutuhkan untuk merepresentasikan nilai keabuan piksel disebut kedalaman piksel (*piksel depth*). Citra sering diasosiasikan dengan kedalaman *piksel*-nya. Jadi, citra dengan kedalaman 8 bit disebut juga citra 8-bit (atau citra 256 warna).

Tabel 2.1. Skala Keabuan dan Rentang Nilai

Skala Keabuan	Rentang Nilai Keabuan	Piksel Dept
$2^1$ (2 nilai)	0, 1	1 bit
$2^2$ (4 nilai)	0 sampai 7	2 bit
$2^3$ (8 nilai)	0 sampai 15	3 bit
$2^8$ (256 nilai)	0 sampai 255	8 bit

## 2.7. Peningkatan Mutu Citra

Peningkatan mutu citra bertujuan untuk memperoleh citra yang dapat memberikan informasi sesuai dengan tujuan atau kepentingan pengolahan citra. Proses peningkatan mutu citra ini termasuk memperbaiki citra yang ketika proses akuisisi mengalami

gangguan yang signifikan seperti *noise*, gangguan geometris, radiometrik, dan beberapa gangguan faktor alam lainnya. Secara umum domain dalam peningkatan mutu citra ini dapat dilakukan secara spasial dan frekuensi. Domain spasial melakukan manipulasi nilai piksel secara langsung dengan dipengaruhi oleh nilai piksel lainnya secara spasial sedangkan domain frekuensi berdasarkan frekuensi spektrum citra. Peningkatan mutu citra dapat dikaitkan dengan metode *filtering*, dimana citra tersebut di-*filter* untuk mendapatkan citra yang lebih baik.

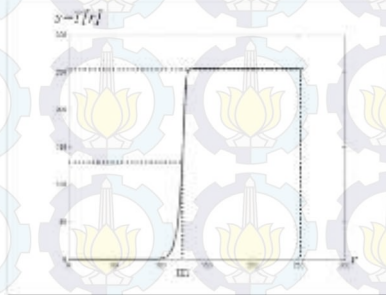
Pada sebuah titik yang memiliki ukuran *neighborhood* adalah  $1 \times 1$ , dimana  $g$  bergantung hanya pada nilai  $f$  di titik  $(x, y)$ .  $T$  yang merupakan fungsi transformasi memetakan  $f$  ke suatu nilai tingkat keabuan tertentu  $s$ , dimana

$$s = T(r) \dots\dots\dots (5)$$

yang dalam hal ini,

$r$  = tingkat keabuan  $f(x, y)$ , dan  $s$  = tingkat keabuan  $g(x, y)$

contoh dari fungsi transformasi intensitas terlihat pada Gambar 2.12.

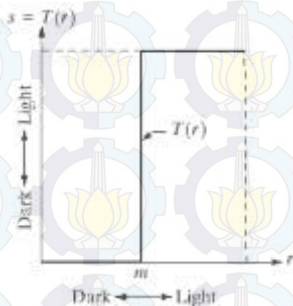


Gambar 2.12. *Contrast-Stretching Function* [6]



*Contrast-Stretching* yaitu mengubah nilai kontras piksel dari piksel aslinya dengan ketentuan menggelapkan piksel dengan nilai tingkat keabuan yang lebih rendah dari  $m$  dan meningkatkan kontras  $t$  apabila nilai tingkat keabuan piksel lebih dari  $m$ .

Selain *Contrast-Stretching*, *thresholding* juga biasa digunakan untuk peningkatan mutu citr. *Thresholding* adalah mentransformasikan citra menjadi citra *biner*, yaitu citra dengan dua warna. Gambar 2.13 menunjukkan bahwa jika nilai tingkat keabuan suatu piksel  $0 \leq r < m$  maka akan ditransformasi ke nilai 0, sebaliknya untuk nilai piksel  $m \leq r \leq (L - 1)$  akan ditransformasikan ke nilai  $L - 1$ .



Gambar 2.13. Transformasi *Threshold* [6]

Hal lain yang bisa dilakukan untuk peningkatan mutu citra yaitu dengan memroses dari histogram citra tersebut. Histogram suatu citra digital dengan suatu tingkat keabuan  $[0, L - 1]$  adalah suatu fungsi diskrit, dengan persamaan sebagai berikut,

$$h(r_k) = n_k \dots \dots \dots (6)$$

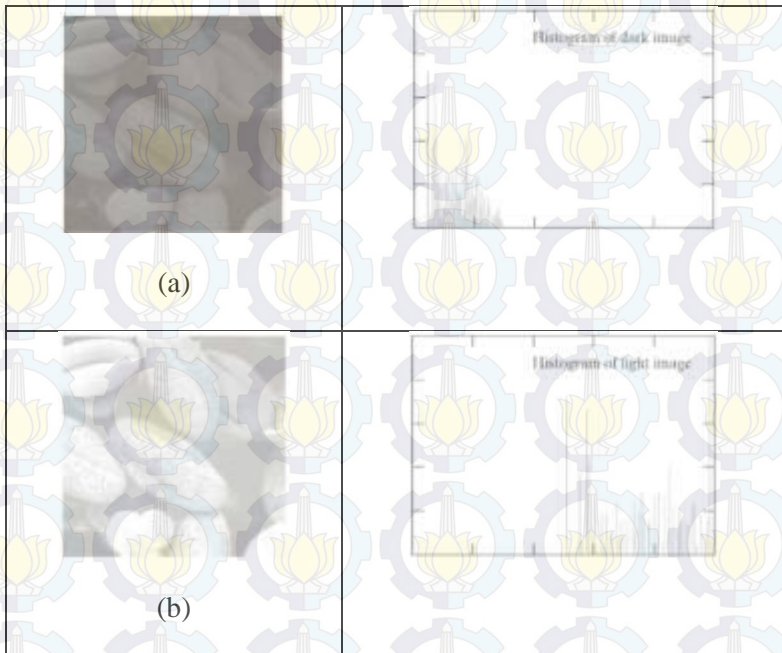
yang dalam hal ini,

$r_k$  = tingkat keabuan ke  $k$

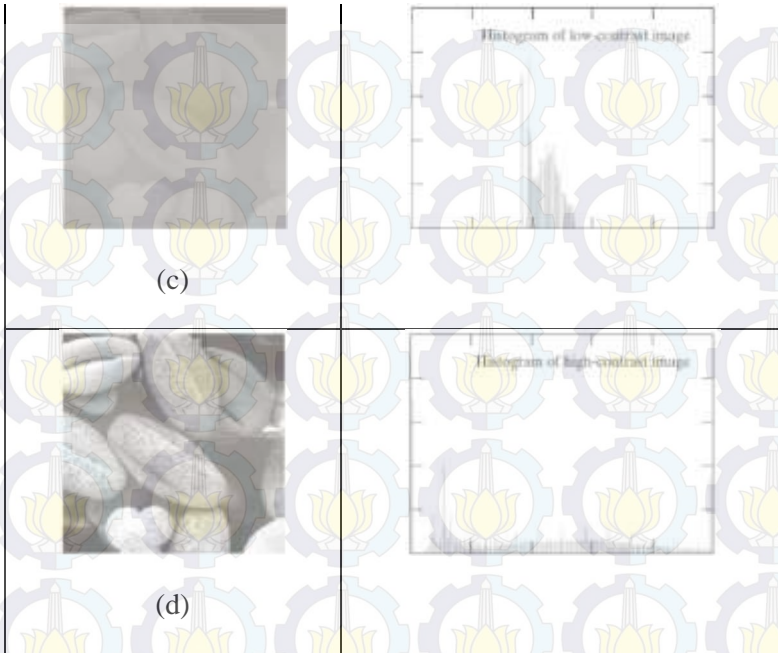
$n_k$  = jumlah total *piksel* dengan tingkat keabuan  $r_k$  pada citra

$h(r_k)$  = histogram citra digital dengan tingkat keabuan  $r_k$ .

Secara umum ada empat tipe dasar dari sebuah citra seperti terlihat pada Gambar 2.14. Dari keempat tipe dasar citra tersebut, perbaikan citra dapat dilakukan yaitu dengan perataan histogram, histogram *matching*, pemrosesan daerah lokal histogram, dan menggunakan statistik dari histogram tersebut [4].



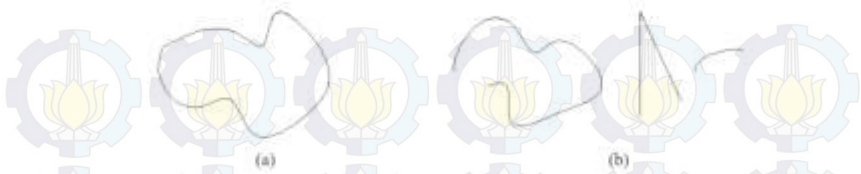
Gambar 2.14. (a). Citra Gelap, (b). Citra Terang



Gambar 2.14. Empat Dasar Tipe Citra dan Histogramnya, (c). Citra Kontras Rendah, dan (d). Citra Kontras Tinggi [4] (Lanjutan)

## 2.8. Kontur dan Representasi

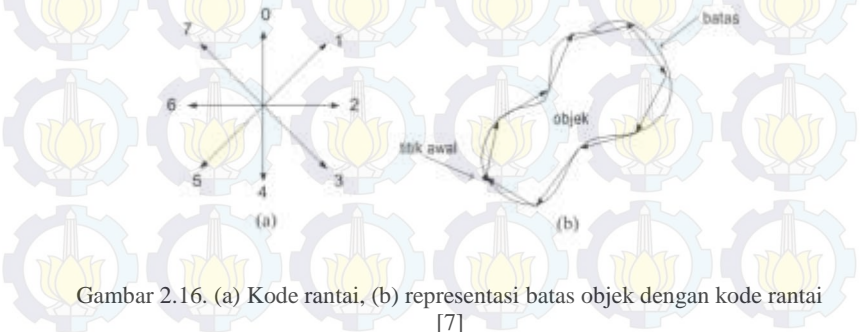
Rangkaian piksel-piksel tepi yang membentuk batas daerah (*region boundary*) disebut kontur (*contour*). Kontur dapat dibedakan menjadi dua yaitu kontur terbuka atau tertutup. Kontur tertutup berkoresponden dengan batas yang mengelilingi suatu daerah lihat Gambar 2.15. Kontur dapat direpresentasi sebagai senarai tepi (*edge list*) atau berupa kurva.



Gambar 2.15.(a) kontur tertutup, (b) kontur terbuka [7]

Representasi kontur ke dalam kurva merupakan representasi yang kompak dan mangkus untuk analisis citra. Terdapat dua cara dalam representasi kontur yaitu dengan kode rantai (*chain code*) dan pencocokan kurva.

Kode rantai (*chain code*) adalah notasi untuk mengkodekan senarai tepi yang membentuk batas daerah. Kode rantai menspesifikasikan arah setiap piksel tepi di dalam senarai tepi. Arah yang digunakan adalah 8 arah mata angin seperti yang terlihat pada pada Gambar 2.16 (a). Dimulai dari sebuah piksel tepi dan searah jarum jam, arah setiap piksel tepi yang membentuk batas objek dikodekan dengan salah satu dari delapan kode rantai. Kode rantai merepresentasikan batas objek dengan koordinat piksel tepi pertama lalu diikuti dengan senarai kode rantai.



Gambar 2.16. (a) Kode rantai, (b) representasi batas objek dengan kode rantai [7]



Ada dua macam teknik pencocokan kurva yaitu interpolasi dan penghampiran (*approximation*). Interpolasi kurva adalah mencari kurva yang melalui semua piksel tepi, sedangkan penghampiran kurva adalah mencari kurva yang paling dekat melalui piksel-piksel tepi tanpa perlu melalui semua piksel tersebut. Salah satu metode penghampiran kurva yang populer dalam pengolahan citra adalah transformasi Hough.

## 2.9. Transformasi Hough Lingkaran

Transformasi hough lingkaran menspesifikasikan kurva dalam bentuk parametrik. Transformasi hough lingkaran menggunakan mekanisme *voting* untuk mengestimasi nilai parameter. Setiap titik di kurva menyumbang suara untuk beberapa kombinasi parameter. Parameter yang memperoleh suara terbanyak terpilih sebagai pemenang. Persamaan lingkaran pada transformasi hough yang berpusat di titik  $(a, b)$  dengan jari-jari  $r$  adalah

$$(x - a)^2 + (y - b)^2 = r^2 \quad \dots\dots\dots (7)$$

Ruang parameter untuk transformasi hough lingkaran adalah  $r-a-b$ , sehingga matriks trimatra  $P(r, a, b)$  dibutuhkan untuk menyimpan perhitungan suara.

Persamaan polar untuk setiap titik  $(x, y)$  di lingkaran:

$$x = a + r \cos q \quad \dots\dots\dots (8)$$

$$y = b + r \sin q \quad \dots\dots\dots (9)$$

Persamaan 8 dan 9 dapat ditulis menjadi persamaan berikut

$$a = x - r \cos q \quad \dots\dots\dots (10)$$

$$b = y - r \sin q \dots\dots\dots (11)$$

Pada operasi deteksi tepi, selain magnitudo piksel tepi, juga dihasilkan arah tepi,  $q$ . Karena itu,  $\cos q$  dan  $\sin q$  dapat dihitung. Misalkan  $(x_i, y_i)$  adalah piksel tepi dan  $q$  adalah arah tepi. Ada dua kasus yang ditinjau yaitu jika jari-jari lingkaran diketahui dan jika jari-jari lingkaran tidak diketahui.

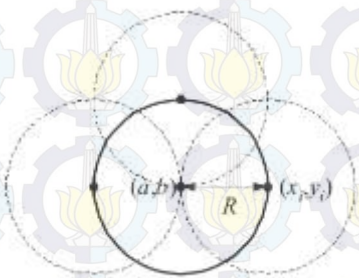
**2.9.1. Jari-jari Lingkaran Diketahui**

Jika jari-jari jari-jari lingkaran diketahui  $r = R$ , maka ruang parametrik trimatra,  $P(r, a, b)$ , dapat direduksi menjadi ruang dwimatra,  $P(a, b)$ .

Titik pusat lingkaran  $(a, b)$  yang mempunyai jari-jari  $r = R$  dan melalui titik  $(x_i, y_i)$  dapat dihitung dengan persamaan

$$a = x_i - R \cos q \dots\dots\dots (12)$$

$$b = y_i - R \sin q \dots\dots\dots (13)$$



Gambar 2.17. Proses Penumpukan *Voting* untuk Mendeteksi Lingkaran [7]

Seperti yang ditunjukkan pada Gambar 2.16, lalu tambahkan elemen  $P(a, b)$  yang bersesuaian dengan satu. Proses ini diulangi

untuk piksel-piksel tepi yang lain. Elemen matriks  $P(a, b)$  yang memiliki jumlah suara di atas nilai ambang tertentu menyatakan lingkaran yang terdapat di dalam citra tepi.

### 2.9.2. Jari-jari Lingkaran tidak Diketahui

Jika jari-jari lingkaran tidak diketahui, maka penumpukan suara dilakukan untuk semua nilai  $r$ ,  $0 < r < r_{max}$ , nilai  $a$  dan  $b$  untuk piksel tepi  $(x_i, y_i)$  dihitung dengan persamaan

$$a = x_i - r \cos q \dots\dots\dots (14)$$

$$b = y_i - r \sin q \dots\dots\dots (15)$$

dan elemen  $P(r, a, b)$  yang bersesuaian dinaikkan satu. Proses ini diulangi untuk piksel-piksel tepi yang lain. Elemen matriks  $P(r, a, b)$  yang memiliki jumlah suara di atas nilai ambang tertentu menyatakan lingkaran yang terdapat di dalam citra tepi.

Persamaan 16 dan 17 dapat dimanipulasi dengan mengeliminasi  $r$  dari kedua persamaan:

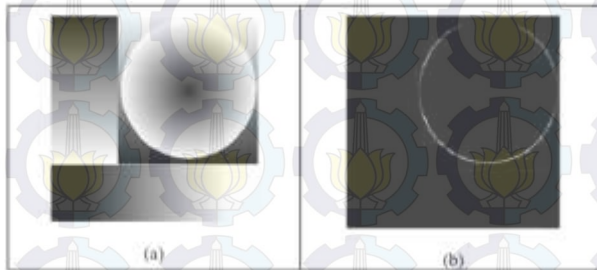
$$a = x - r \cos q \rightarrow r = \frac{(x-a)}{\cos\theta}$$

$$b = y - r \sin q \rightarrow r = b - \frac{(x-a)}{\cos\theta} \sin\theta = y - (x-a)\tan\theta$$

$$b = a \tan\theta - x \tan\theta + y \dots\dots\dots (16)$$

Dengan demikian, maka ruang parametrik trimatra  $P(r, a, b)$  dapat direduksi menjadi ruang dwimatra  $P(a, b)$ . Untuk semua nilai  $r$ , yang dalam hal ini  $a_1 < a < a_k$  nilai ordinat  $b$  dari titik pusat lingkaran  $(a, b)$  yang melalui titik  $(x_i, y_i)$  dapat dihitung dengan persamaan (11), lalu tambahkan elemen  $P(a, b)$  yang bersesuaian dengan satu. Proses ini diulangi untuk piksel-piksel

tepi yang lain. Elemen matriks  $P(a, b)$  yang memiliki jumlah suara di atas nilai ambang tertentu menyatakan lingkaran yang terdapat di dalam citra tepi. Gambar 2.18 memperlihatkan hasil transformasi Hough untuk mendeteksi lingkaran dari citra slope dengan menggunakan nilai ambang  $T = 30$ .



Gambar 2.18. (a) Citra slope, (b) hasil deteksi lingkaran dengan Transformasi Hough [7]

## 2.10. Evaluasi Kemampuan Pendeteksiian Sistem

Evaluasi sistem dilakukan untuk mengetahui seberapa handal sistem yang telah dibuat dengan mengukur nilai kemampuan pendeteksiannya. Sebelum melakukan perhitungan, tentukan terlebih dahulu jumlah data yang terklarifikasi benar dan yang terklarifikasi salah. Jumlah data yang terklarifikasi benar adalah jumlah data hasil klarifikasi yang memiliki prediksi kelas yang sesuai dengan data sebenarnya (pengamatan manual oleh tenaga ahli). Jumlah data yang terklarifikasi salah adalah jumlah data hasil klarifikasi yang memiliki prediksi kelas yang tidak sesuai dengan data sebenarnya. Nilai akurasi dapat didapatkan dengan persamaan 19 sebagai berikut ini.

$$\text{Kemampuan Pendeteksiian} = \frac{\text{Jumlah Data Terklarifikasi Benar}}{\text{Total Data}} \dots (17)$$



## 2.11. OpenCV

OpenCV adalah sebuah *library open source* untuk visi komputer yang bisa didapatkan dari <http://SourceForge.net/projects/opencvlibrary>. *Library* ini ditulis dengan bahasa C dan C++, serta dapat dijalankan dengan Linux, Windows, dan Mac OS X. OpenCV dirancang untuk efisiensi komputasional dan dengan fokus yang kuat pada aplikasi *real-time*.

Salah satu tujuan OpenCV adalah untuk menyediakan infrastruktur visi komputer yang mudah digunakan yang membantu orang-orang dalam membangun aplikasi-aplikasi visi yang *sophisticated* dengan cepat. *Library* pada OpenCV berisi lebih dari 500 fungsi yang menjangkau berbagai area dalam permasalahan visi, meliputi inspeksi produk pabrik, pencitraan medis, keamanan, antarmuka pengguna, kalibrasi kamera, visi stereo, dan robotika. Karena visi komputer dan pembelajaran mesin seringkali berkaitan, OpenCV juga memiliki *Machine Learning Library* (MLL). *Sublibrary* ini berfokus pada pengenalan pola statistik dan *clustering*. MLL sangat berguna untuk tugas-tugas visi yang berada dalam misi inti OpenCV, tetapi MLL cukup umum digunakan untuk permasalahan pembelajaran mesin.

Lisensi *open source* pada OpenCV telah distrukturisasi sehingga pengguna dapat membangun produk komersial menggunakan seluruh bagian pada OpenCV. Tidak ada kewajiban untuk meng-*open source* produk tersebut atau untuk memberikan peningkatan ke domain publik. Sebagian karena peraturan lisensi liberal ini, maka terdapat komunitas pengguna dalam jumlah yang sangat besar, termasuk di dalamnya orang-orang dari perusahaan besar (seperti IBM, Microsoft, Intel, SONY, Siemens, dan Google) serta pusat-pusat penelitian (seperti Stanford, MIT, CMU, Cambridge, dan INRIA).

Sejak peluncuran pertamanya pada Januari 1999, OpenCV telah digunakan pada banyak aplikasi, produk, dan usaha-usaha penelitian. Aplikasi-aplikasi ini meliputi penggabungan citra pada peta web dan satelit, *image scan alignment*, pengurangan *noise* pada citra medis, sistem keamanan dan pendeteksian gangguan, sistem pengawasan otomatis dan keamanan, sistem inspeksi pabrik, kalibrasi kamera, aplikasi militer, serta kendaraan udara tak berawak, kendaraan darat, dan kendaraan bawah air.

OpenCV adalah singkatan dari *Open Computer Vision*, yaitu *library open source* yang dikembangkan untuk melakukan pengolahan citra. Tujuannya adalah agar komputer mempunyai kemampuan yang mirip dengan cara pengolahan visual pada manusia. *Library* ini dibuat untuk bahasa C/C++ sebagai optimasi aplikasi *real-time*. OpenCV memiliki API (*Application Programming Interface*) untuk pengolahan tingkat tinggi maupun tingkat rendah. Pada OpenCV juga terdapat fungsi-fungsi siap pakai untuk me-load, menyimpan, serta mengakuisisi gambar dan video.

*Library* OpenCV (7) memiliki fitur-fitur sebagai berikut:

- Manipulasi data gambar (mengalokasi memori, melepaskan memori, menduplikasi gambar, mengatur serta mengkonversi gambar)
- *Image/Video I/O* (bisa menggunakan kamera yang sudah didukung oleh library ini)
- Manipulasi matriks dan vektor, serta terdapat juga routines aljabar linear (*products, solvers, eigenvalues, SVD*)
- Pengolahan citra dasar (penapisan, pendeteksian tepi, sampling dan interpolasi, konversi warna, operasi morfologi, histogram, piramida citra)
- Analisis struktural

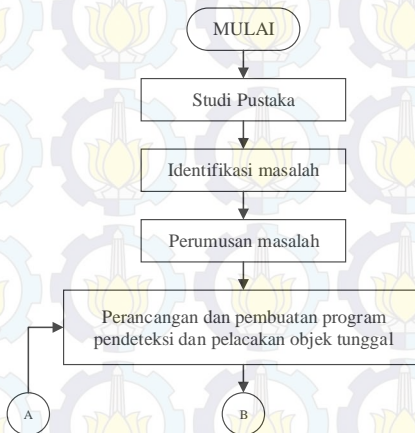
- Kalibrasi kamera
- Pendeteksian gerakan
- Pengenalan objek
- GUI dasar (menampilkan gambar dan video, mengontrol *mouse* atau *keyboard*, *scrollbar*)
- *Image labelling* (garis, kerucut, poligon, penggambaran teks)

*Libraries* OpenCV menyediakan banyak algoritma visi komputer dasar, dengan keuntungan bahwa fungsi-fungsi tersebut telah diuji dengan baik dan digunakan oleh para peneliti di seluruh dunia. *Libraries* OpenCV juga menyediakan sebuah modul untuk pendeteksian objek yang menggunakan algoritma Viola Jones.

## BAB III METODOLOGI

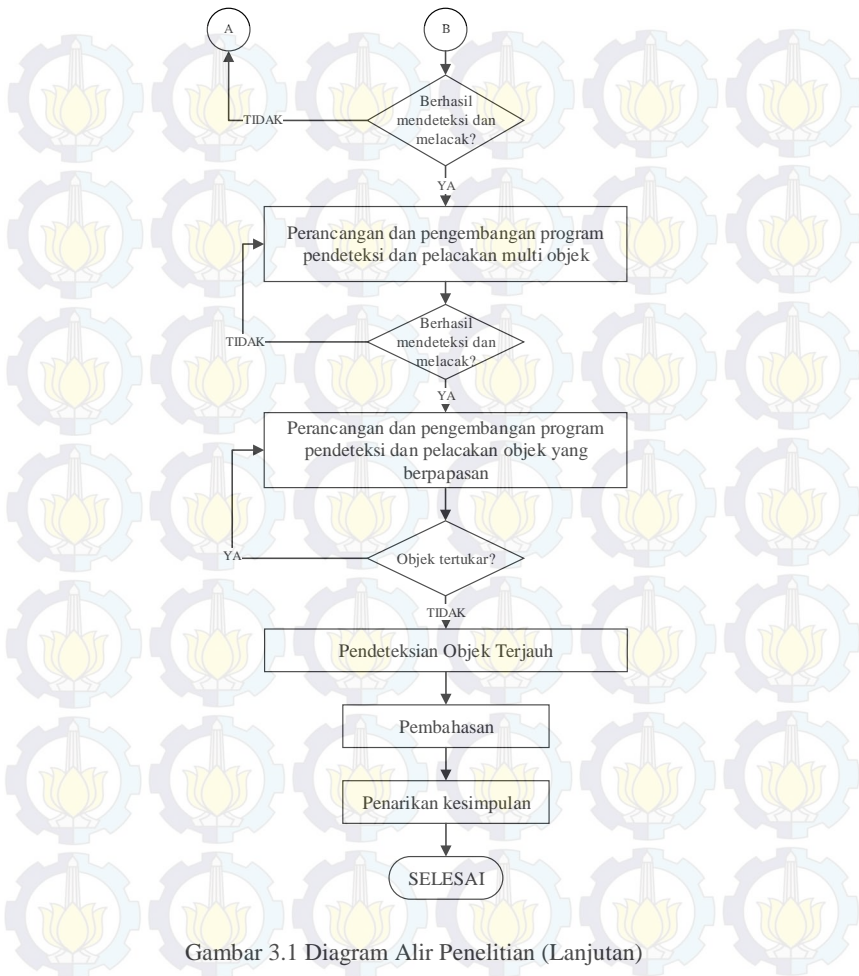
Metodologi penelitian ini menggambarkan langkah-langkah yang nantinya akan dilakukan dalam penelitian ini. Tujuan dari pembuatan metodologi penelitian ini yaitu agar proses dalam penelitian ini terstruktur dengan baik dan dapat mencapai sasaran. Metodologi penelitian yang digunakan dalam penelitian ini secara garis besar dapat dibagi menjadi beberapa tahapan, yaitu tahap studi pustaka, tahap identifikasi masalah, tahap perumusan masalah, tahap perancangan program, tahap pengumpulan dan pengolahan data serta tahap penarikan kesimpulan. Selain itu juga akan dijelaskan spesifikasi alat dan parameter proses yang digunakan dalam penelitian ini. Adapun tahapan-tahapan yang dilakukan dalam proses penelitian ini seperti diagram alir pada Gambar 3.1.

### 3.1. Diagram Alir Penelitian



Gambar 3.1 Diagram Alir Penelitian





### 3.2. Kajian Pustaka

Studi kepustakaan diperlukan sebagai acuan referensi penulis untuk mendalami permasalahan yang akan diteliti mengenai pendeteksian dan pelacakan objek tunggal maupun multi

objek dari buku-buku referensi dan jurnal yang berkaitan dengan permasalahan yang akan dibahas.

### **3.3. Identifikasi Masalah**

Identifikasi masalah dilakukan dengan menyusun sistem pendeteksian dan pelacakan objek tunggal dan multi objek.

### **3.4. Perumusan Masalah**

Perumusan masalah mencakup perancangan program untuk mendeteksi dan melacak objek tunggal dan multi objek.

### **3.5. Perancangan dan Pengembangan Program**

Perancangan dan pengembangan program pendeteksian lebih lanjut akan dibahas pada Bab 4.

#### **3.5.1. Inisialisasi Program**

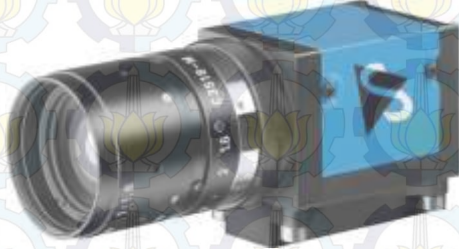
Ada beberapa hal yang perlu diperhatikan dalam tahap inisialisasi program. Diantarnya sebagai berikut

1. Objek yang dideteksi adalah objek dengan bentuk bulat seperti ditunjukkan pada Gambar 3.3 berikut ini



Gambar 3.2. Objek Pendeteksian dan Pelacakan

2. Pengolahan data menggunakan notebook.
3. Menggunakan kamera usb yang terlihat pada Gambar 3.4 dan spesifikasi dapat dilihat pada Lampiran 1.



Gambar 3.3. Kamera usb The Imaging Source DFK31BU03

### 3.5.2. Tahapan Proses

Secara umum dalam penelitian ini akan dilakukan dalam beberapa tahapan proses sebagai berikut,

#### 1. Persiapan

Pada tahap ini, ada beberapa hal yang dipersiapkan diantaranya pembuatan tampilan window untuk gambar asli, gambar segmentasi, gambar hasil, dan posisi objek terdeteksi.

#### 2. *Preprocessing*

Proses ini diawali proses segmentasi objek dari latar belakangnya yaitu dengan menentukan *depth threshold* untuk menentukan wilayah objek berada. Hal ini digunakan untuk mengantisipasi kesalahan dalam menentukan objek, sebagai contoh segmentasi dengan menggunakan *skin filter* terlihat pada Gambar 3.4. Pada Gambar (a) merupakan citra sebelum segmentasi sedangkan Gambar (b) merupakan citra setelah dilakukan segmentasi.



Gambar 3.4. Segmentasi Objek

Segmentasi dengan menggunakan skin filter yang sangat dipengaruhi dengan masalah pencahayaan. Karena sangat terpengaruh dengan cahaya, segmentasi didasarkan berdasarkan pada keadaan histogram dari objek. Dari histogram tersebut akan didapatkan range warna yang bisa digunakan untuk pelacakan objek pada *frame* berikutnya. Lebih lanjut akan dijelaskan pada Bab IV.

### 3. Mendeteksi objek

Pada pendeteksi objek ini dilakukan dengan pendekatan kontur, yaitu mencari kontur dari objek melalui *binary image*. Kontur yang diambil adalah kontur tepi dari objek linkaran. Contoh hasil pendeteksian terlihat pada Gambar 3.5. Pencarian tepi objek dilakukan terhadap bidang 2D karena pencarian kontur dilakukan dalam bidang 2D.

### 4. Pelacakan objek

Pelacakan objek dilakukan dengan metode *particle filter* berdasarkan histogram warna RGB. Hasil yang didapat dari pelacakan ini akan diolah untuk mendapatkan tingkat *error* dan



kemampuan pendeteksian dari program. Kemampuan pendeteksian dihitung dari perbandingan objek yang berhasil dideteksi dengan jumlah objek yang ada.

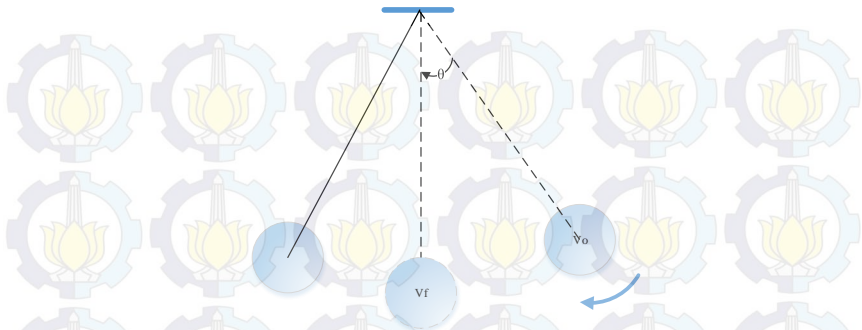
### 3.6. Pembahasan

Adapun beberapa percobaan yang dilakukan dalam penelitian ini, diantaranya sebagai berikut

1. Mendeteksi keberadaan objek tidak bergerak (diam) dalam kondisi 1 dan kondisi 2.
2. Mendeteksi dan melacak pergerakan satu buah objek dengan beberapa macam kecepatan dalam kondisi 1 dan kondisi 2.
3. Mendeteksi dan melacak pergerakan satu buah bola yang bergerak menjauh.
4. Mendeteksi dan melacak pergerakan multi objek yang saling bersilangan.

Kondisi 1 dilakukan pada malam hari dengan dibantu pencahayaan lampu dan kondisi 2 pada siang hari dengan tidak menerima cahaya matahari secara langsung.

Variasi kecepatan untuk objek tunggal dilakukan dengan menggantungkan objek dengan seutas tali seperti terlihat pada Gambar 3.5 dengan panjang tali adalah 0,35 meter. Dari masing-masing percobaan akan didapatkan *error* yang terjadi dari setiap *frame*. Selain itu juga akan dianalisa kemampuan pendeteksian dari program yang digunakan dengan menganalisa hasil deteksi.



Gambar 3.5. Pengantungan Objek dengan Seutas Tali

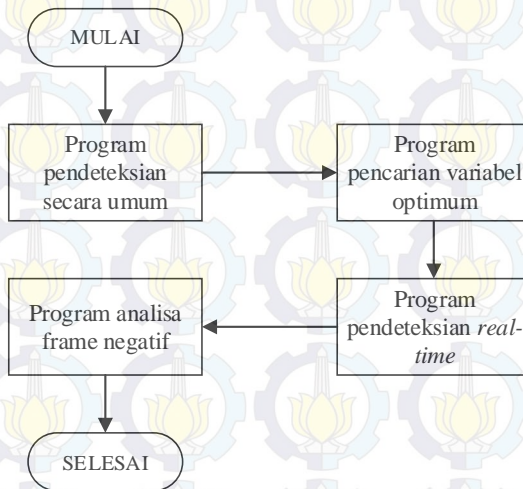
### 3.7. Kesimpulan dan Saran

Dari keempat percobaan yang dilakukan akan didapatkan suatu kesimpulan dan saran serta rekomendasi untuk penelitian lebih lanjut.



## BAB IV PERCOBAAN

Proses pendeteksian objek seperti terdapat pada diagram alir perancangan program Gambar 4.1.



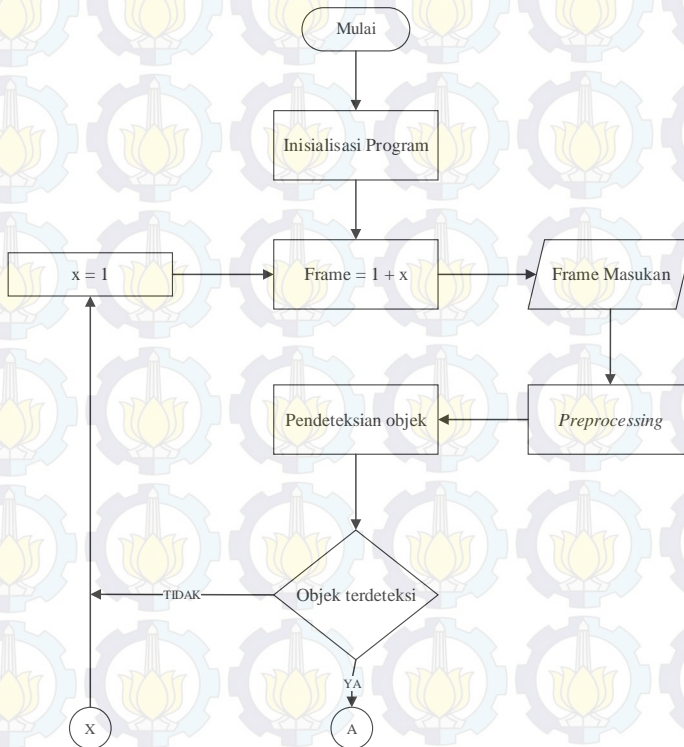
Gambar 4.1. Diagram Alir Proses Perancangan Program

Gambar 4.1 tersebut, secara umum proses perancangan program dibagi menjadi empat proses yaitu pembuatan program pendeteksian objek, program pencarian variabel, program pendeteksian secara *real-time*, dan program untuk analisa *frame* negatif (*frame* kesalahan pendeteksian objek). Setiap proses memiliki peranan masing-masing.

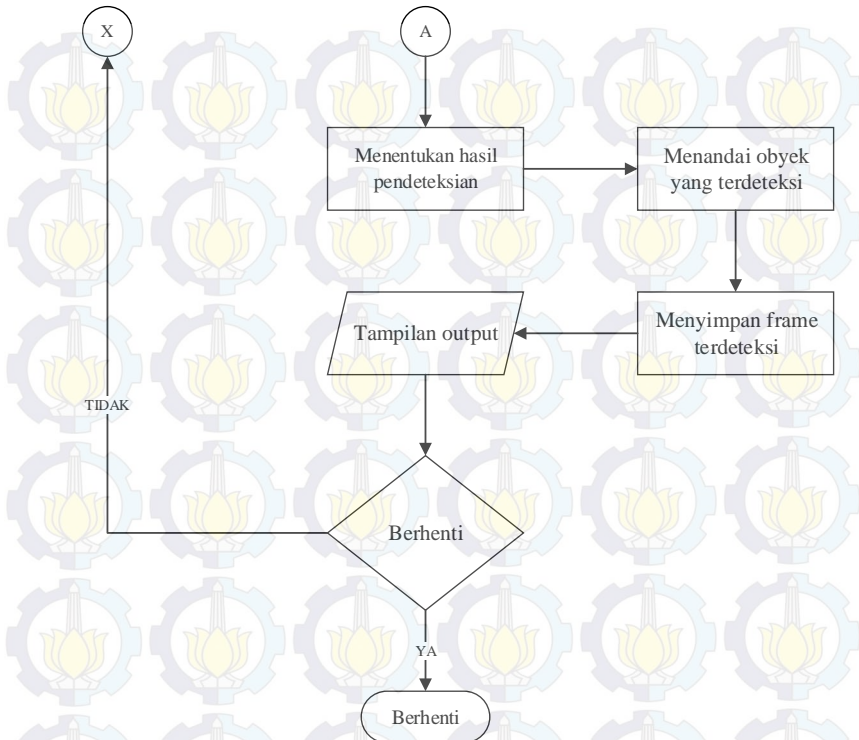


#### 4.1. Program Pendeteksian Objek

Diagram alir program pendeteksian objek terlihat seperti pada diagram alir Gambar 4.2.

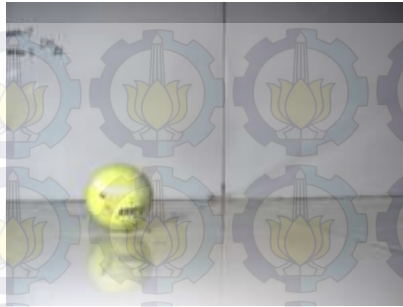


Gambar 4.2. Diagram Alir Program Pendeteksian



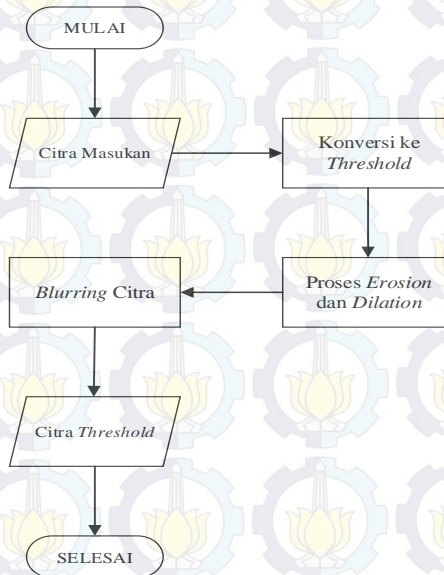
Gambar 4.2. Diagram Alir Program Pendeteksian (Lanjutan)

*Frame* hasil dari *capture* kamera USB berbentuk dalam format citra RGB (*Red Gren Blue*) 8-bit dengan resolusi cirtra 640 x 480 piksel. Hasil dari *frame* tersebut seperti terlihat pada Gambar 4.3.

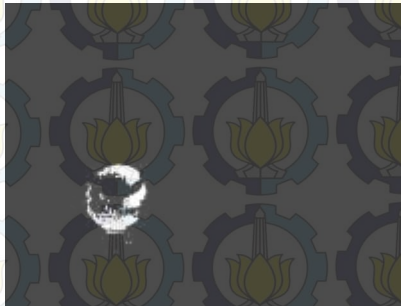


Gambar 4.3. Citra Asli

Citra format RGB tersebut, selanjutnya akan dilakukan proses *preprocessing*. Digambarkan dari proses *preprocessing* seperti terlihat pada Gambar 4.4.

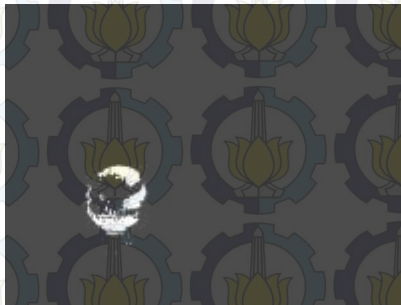
Gambar 4.4. Diagram Alir Proses *Preprocessing*

Citra format RGB selanjutnya dirubah ke dalam format citra *threshold*. Tipe *threshold* yang digunakan adalah *binary threshold* dengan *range* nilai RGB minimum ( 0, 0, 0) – maksimum ( 255, 255, 255) yang merupakan citra RGB 8-bit yaitu 256. Hasil perubahan citra RGB ke format *threshold* terlihat seperti pada Gambar 4.5.



Gambar 4.5. Citra *Threshold*

Citra *threshold* kemudian dilakukan *erode*. *Erode* dilakukan dengan pengecilan 2x2 piksel. Hasil *erode* terlihat seperti pada Gambar 4.6.



Gambar 4.6. Citra *Threshold* Hasil *Erode* 2x2 piksel

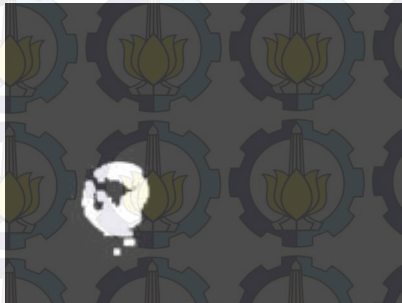


Citra *threshold* yang sudah di-*erode* kemudian dilakukan *dilate* dengan ukuran 3x3 piksel. Hasil dari *dilate* tersebut terlihat seperti pada Gambar 4.7.



Gambar 4.7. Citra *Threshold* Hasil dilade 9x9 Piksel

Citra *threshold* yang sudah dilakukan *erode* dan *dilate* selanjutnya dilakukan *smoothing*. *Smoothing* pada format ini difungsikan untuk mempermudah pendeteksian supaya hasil pendeteksian sesuai yang diharapkan. Penge-*blur*-an dilakukan dengan kernel size 3x3 dan hasil seperti terlihat pada Gambar 4.8.



Gambar 4.8. Hasil *Smoothing* Citra *Threshold*

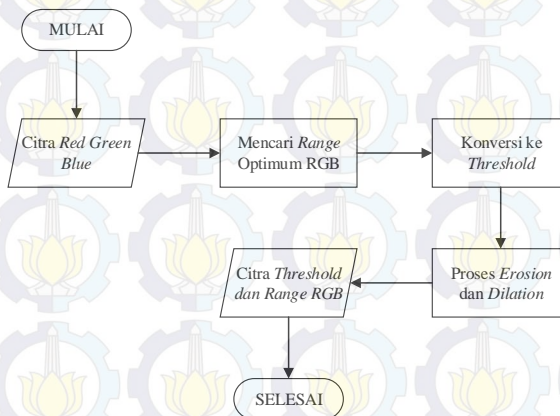
Pendeteksian objek menggunakan `CV_HOUGH_CIRCLES` yang merupakan fungsi dari *library* `opencv`. Variabel dari `CV_HOUGH_CIRCLES` tersebut antara lain inverse rasio resolusi, jarak minimum antar pusat lingkaran, parameter 1, parameter 2, radius minimum lingkaran, dan radius maksimum lingkaran. Variabel tersebut akan dicari dengan menggunakan program pencarian variabel.

Objek terdeteksi mempunyai informasi diantaranya koordinat pusat objek terdeteksi (x,y) dan radius objek terdeteksi dalam piksel. Penandaan lokasi objek dilakukan dengan memakai *rectangle* dengan warna *rectangle* didasarkan pada warna citra *threshold* yang sebelumnya merupakan citra RGB.

## 4.2. Pencarian Nilai Variabel Pendeteksian

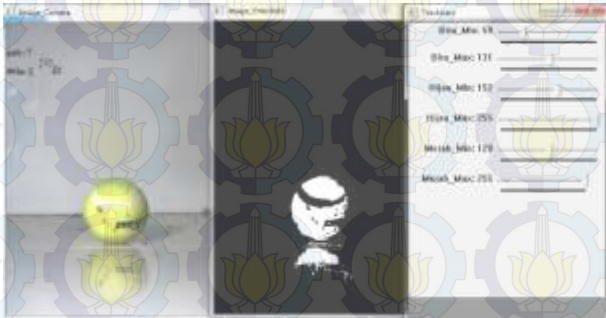
### 4.2.1. Variabel *Range* Warna *Threshold*

Diagram pencarian *range* warna untuk merubah citra RGB ke citra *threshold* dilakukan seperti Gambar 4.9.



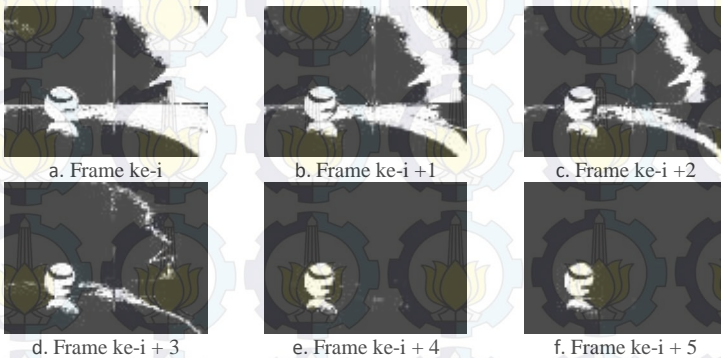
Gambar 4.9. Diagram Alir Pencarian *Range* Warna

Hasil citra RGB kemudian dilakukan pencarian *range* citra RGB perubahan ke citra *threshold* seperti pada Gambar 4.10.

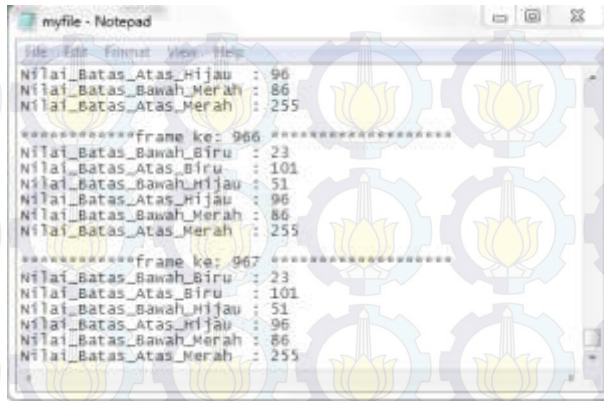


Gambar 4.10. *Trackbar* Pencari Perubahan Citra RGB ke Citra *Threshold*

Pencarian *range* tersebut akan diperoleh perubahan *threshold* dengan menggunakan *trackbar* seperti pada Gambar 4.11 dan informasi seperti terlihat pada Gambar 4.12.



Gambar 4.11. Citra *Threshold* dari *Trackbar*



Gambar 4.12. Informasi Hasil Pencarian *Range* dengan *trackbar*

Hasil dari *range* yang didapatkan kemudian digunakan untuk poses pendeteksian objek. Hasil terbaik dilihat dari citra *threshold* terbaik yang telah tersimpan pada setiap framenya.

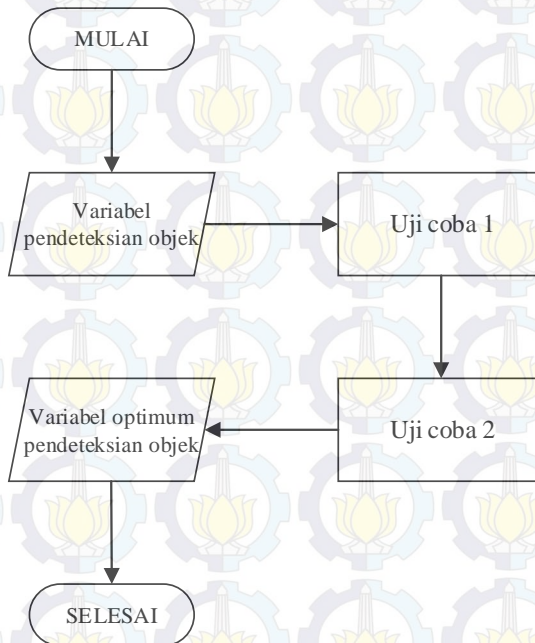
#### 4.2.2. Pencarian Variabel Pendeteksian Objek

Dalam pendeteksian lingkaran, fasilitas yang digunakan untuk mendeteksi lingkaran yaitu dengan menggunakan `CV_HOUGH_CIRCLE` dari OpenCV. Variabel dari `CV_HOUGH_CIRCLE` tersebut antara lain inverse rasio resolusi citra, jarak minimum pusat lingkaran, parameter 1, parameter 2, radius minimum lingkaran dan radius maksimum lingkaran. Pendeteksian ini digunakan untuk mendeteksi multi objek benda bulat dengan kemungkinan ada benda yang berpapasan sehingga ada variabel yang ditetapkan sebagai berikut

- Jarak minimum pusat lingkaran yang digunakan 20 piksel
- Radius minimum 1 piksel
- Radius maksimum 200 piksel



Sedangkan untuk invers rasio resolusi citra, parameter 1, dan parameter 2 akan dicari nilai variabel dengan melakukan percobaan dengan menguji pada sejumlah *frame* pada masing-masing objek yang akan dideteksi. Pencarian variabel tersebut seperti yang ditunjukkan pada diagram alir Gambar 4.13. Parameter 1 adalah batas *threshold* untuk deteksi pusat dan parameter 2 batas *threshold* untuk mendeteksi *canny*.



Gambar 4.13. Diagram Alir Pencarian Variabel

Skenario uji coba yang dilakukan untuk mencari variabel adalah sebagai berikut.

1. Uji coba 1. Uji coba ini bertujuan untuk mengetahui pengaruh variabel inverse rasio resolusi, parameter 1, dan parameter 2.
2. Uji coba 2. Uji coba ini bertujuan untuk mengetahui variabel dari inverse rasio resolusi citra, parameter 1, dan parameter 2.

Masing-masing uji coba dilakukan dengan *trial* dan *error* dimana hasil terbaik dilihat dari hasil yang memiliki keberhasilan paling tinggi dan juga *error* paling rendah.

#### 4.2.2.1. Uji Coba 1

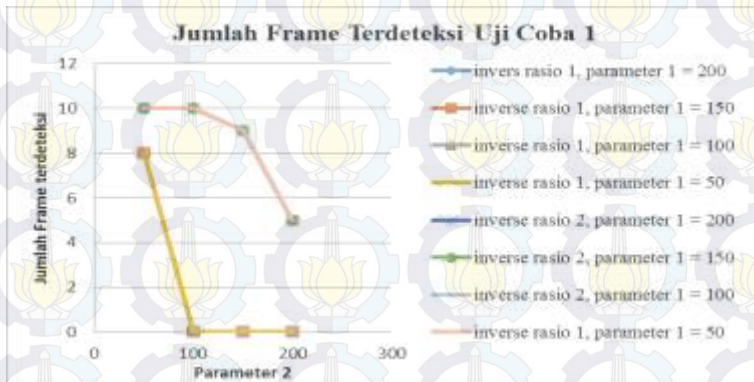
Pada uji coba ini digunakan masukan berupa citra yang terdapat objek berbentuk lingkaran di dalamnya. Uji coba ini bertujuan untuk mengetahui pengaruh dari variabel pendeteksian lingkaran invers rasio resolusi, parameter 1, dan parameter 2 dalam pendeteksian dari keberadaan objek berbentuk lingkaran. Objek lingkaran yang digunakan untuk uji coba 1 seperti terlihat pada Gambar 4.14.



Gambar 4.14. Objek Uji Coba 1[8]

Hasil uji coba 1 menggunakan inputan sebanyak 10 *frame* seperti pada Lampiran 2 dengan kondisi yang tidak sama

tetapi masih mempunyai satu buah objek yang sama diperoleh hubungan seperti terlihat pada Gambar 4.15.



Gambar 4.15. Hubungan Inverse Rasio Resolusi, Parameter 1, dan Parameter 2 terhadap Proses Pendeteksian

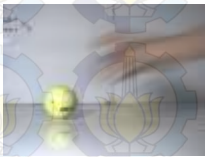
Dari grafik tersebut terlihat bahwa kemungkinan terdeteksi lebih besar ketika inverse rasio resolusi citra lebih besar dan parameter 2 semakin kecil. Hal tersebut terlihat ketika invers rasio resolusi sebesar 2 dan parameter 2 sebesar 200 sudah bisa mendeteksi keberadaan objek sebanyak 5 *frame* dari 10 *frame* masukan. Sedangkan dengan invers rasio resolusi sebesar 1 dan parameter 2 sebesar 200 masih belum bisa mendeteksi keberadaan objek. Untuk besaran pada parameter 1 tidak terlalu berpengaruh terhadap proses pendeteksian.

#### 4.2.2.2. Uji Coba 2

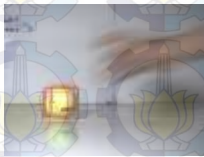
Pada uji coba ini, inputan yang digunakan adalah inputan dari *frame* objek sebanyak 10 *frame*. Uji coba 2 dilakukan dengan menggunakan variasi dari parameter 1 dan parameter 2. Variasi dari parameter 1 yaitu 50, 100, 150, 200, dan 250. Sedangkan

variasi dari parameter 2 yaitu antara 50 sampai 200 dengan kenaikan angka sebesar 5.

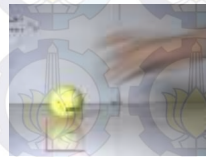
Hasil dari pendeteksian ini akan dianalisa berdasarkan hasil *frame* keluaran. Hasil tersebut akan dibedakan menjadi empat jenis *frame*. *Frame* positif digunakan jika pada *frame* tersebut pendeteksian berada di dalam objek sedangkan *frame* negatif digunakan untuk kesalahan pendeteksian. *Frame* negatif (kesalahan pendeteksian) dibedakan menjadi tiga jenis diantaranya negatif tipe 1, tipe 2, dan tipe 3 seperti terlihat pada Gambar 4.16. Tipe 1 digunakan jika objek yang seharusnya terdeteksi tetapi tidak terdeteksi, tipe 2 digunakan jika jumlah pendeteksian pada objek lebih dari jumlah objek, sedangkan tipe 3 digunakan jika pendeteksian berada di luar keberadaan objek. Hasil dari uji coba 2 seperti yang terlampir pada Lampiran 4.



a. Kesalahan Pendeteksian Tipe 1



b. Kesalahan Pendeteksian Tipe 2



c. Kesalahan Pendeteksian Tipe 2

Gambar 4.16. Tipe Kesalahan Pendeteksian Objek





## BAB V PEMBAHASAN

Pada bab ini akan dibahas hasil data pencarian variabel *range* warna, pendeteksian objek tunggal, dan pendeteksian multi objek yang telah didapatkan kemudian akan dianalisa dari data yang diperoleh tersebut.

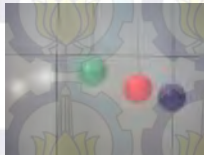
### 5.1. Pencarian *Range* Variabel Pendeteksian

Hasil dari pencarian *range* RGB (*Red*(merah), *Green*(Hijau), *Blue*(Biru)) seperti terlihat pada Tabel 5.1.

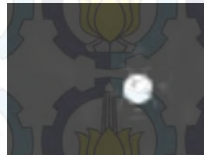
Tabel 5.1. Hasil Pencarian *Range* RGB

No.	Objek	Merah		Hijau		Biru	
		Min.	Mak.	Min.	Mak.	Min.	Mak.
1	Merah	160	255	0	75	0	90
2	Hijau	40	90	100	255	60	180
3	Ungu	30	125	30	70	50	150

*Range* warna RGB tersebut didapatkan dari objek seperti terlihat pada Gambar 5.1.

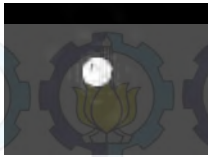


a. Objek Asli



b. *Threshold* Objek Merah

Gambar 5.1. Objek Asli dan Hasil *Threshold*

c. *Threshold Objek Hijau*d. *Threshold Objek Ungu*Gambar 5.1. Objek Asli dan Hasil *Threshold* (Lanjutan)

Dari ketiga objek tersebut didapatkan jarak antara batas jarak warna minimum dan maksimum dari masing-masing objek seperti pada Tabel 5.2.

Tabel 5.2. Jarak Warna RGB Objek

No.	Objek	Merah	Hijau	Biru	Jumlah Kombinasi
1	Merah	95	75	90	<u>641.250</u>
2	Hijau	50	155	120	<u>775.000</u>
3	Ungu	95	40	100	<u>380.000</u>

Kombinasi warna primer (merah, hijau, biru) terbanyak dimiliki objek warna hijau dengan 775.000 kombinasi, kemudian objek merah dengan 641.250 kombinasi dan kombinasi paling sedikit dimiliki objek ungu dengan 380.000 kombinasi warna RGB.

Dengan membagi warna RGB 8-bit (0, 255) menjadi dua bagian terpisah dengan nilai tengah sebesar 125 maka akan didapatkan *range* yang berada di *range* warna bagian atas dan bagian bawah. *Range* tersebut seperti ditunjukkan pada Tabel 5.3.

Tabel 5.3. *Range* Bagian Atas dan Bagian Bawah

No.	Objek	Merah		Hijau		Biru		Total	
		Bawah	Atas	Bawah	Atas	Bawah	Atas	Bawah	Atas
1	Merah	0	95	75	0	90	0	<u>165</u>	<u>95</u>
2	Hijau	50	0	25	130	65	55	<u>140</u>	<u>185</u>
3	Ungu	95	0	40	0	75	25	<u>210</u>	<u>25</u>

Dalam *range* warna 8-bit tersebut, warna yang mendekati nilai terkecil akan cenderung gelap dan sebaliknya apabila warna tersebut mendekati nilai terbesar akan cenderung memiliki warna terang. Dari Tabel 5.3 tersebut, objek merah memiliki perbandingan lebih banyak bagian bawah dari pada bagian atas dengan perbandingan 165:95. Objek hijau memiliki perbandingan lebih banyak bagian atas dibandingkan bagian bawah dengan perbandingan sebesar 140:185. Sedangkan objek ungu memiliki perbandingan lebih banyak bagian bawah dari pada bagian atas dengan perbandingan sebesar 210:25.

Besar perbandingan warna bagian atas dan bawah akan mempengaruhi pendeteksian objek apabila terdapat objek dengan warna tertentu terdapat pada *frame*. Yang paling menonjol perbandingan warna adalah pada objek ungu dimana bagian bawah 8 kali lebih banyak dari bagian atas. Hal ini bisa menyebabkan apabila terdapat objek cenderung gelap dan berebentuk lingkaran akan terdeteksi sebagai objek ungu.

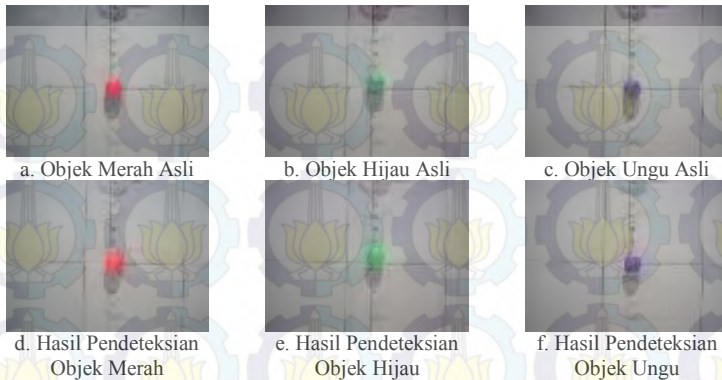
## **5.2. Objek Tunggal**

Pendetksian objek tunggal dilakukan pada setiap objek yaitu bola dengan warna dominan merah, hijau, dan ungu. Pendeteksian dilakukan pada dua kondisi berbeda yaitu pada kondisi 1 dan kkondisi 2.

### **5.2.1. Kemampuan Pendeteksian Objek Diam**

Objek yang digunakan untuk mencari kemampuan pendeteksian objek dalam kondisi diam seperti terlihat pada Gambar 5.2.





Gambar 5.2. Hasil Pendeteksian Objek Diam Kondisi 1

Pendeteksian objek diam pada kondisi 1 dengan jumlah *frame* sebanyak 1000 *frame* diperoleh hasil seperti pada Tabel 5.4.

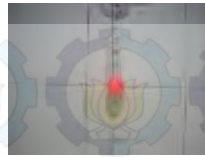
Tabel 5.4. Hasil Pendeteksian Objek Diam Kondisi 1

No.	Objek	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan pendeteksian (%)
			Tipe 1	Tipe 2	Tipe 3	
1	Merah	997	3	0	0	99,7 %
2	Hijau	957	43	0	0	95,7 %
3	Ungu	996	4	0	0	99,6 %

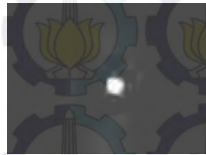
Kemampuan pendeteksian tertinggi dimiliki objek merah dengan kemampuan pendeteksian sebesar 99,7 % diikuti objek ungu dengan 99,6 %, dan terendah dimiliki objek hijau dengan kemampuan pendeteksian sebesar 95,7 %. Hal tersebut terjadi dikarenakan pada kondisi 1 dimana dilakukan pada malam hari, *range* warna dengan warna dominan gelap akan mudah dideteksi daripada warna dengan dominan terang. Keseluruhan objek tidak terdeteksi adalah objek tidak terdeteksi tipe 1. Contoh perbandingan hasil *preprocessing* objek merah seperti terlihat pada Gambar 5.3.



a. Objek Terdeteksi Asli

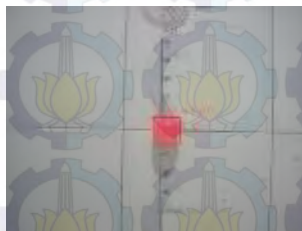


b. Objek Tidak Terdeteksi Asli

c. Objek Terdeteksi Citra *Threshold*d. Objek Tidak Terdeteksi Citra *Threshold*

Gambar 5.3. Perbandingan Hasil *Preprocessing* Objek Merah Terdeteksi dan Objek Merah Tidak Terdeteksi

Citra *threshold* hasil perbandingan tidak memiliki perbedaan yang signifikan. Dengan melakukan evaluasi pendeteksian terhadap *frame* tersebut, didapatkan bahwa *frame* tersebut bisa dideteksi sehingga seharusnya *frame* tersebut bisa dideteksi secara *real-time*. Hasil evaluasi pendeteksian objek tidak terdeteksi tersebut didapatkan hasil bahwa objek berada pada koordinat  $x = 337$  dengan  $y = 261$  dengan radius terdeteksi sebesar 28 piksel. Hasil pendeteksian tersebut terlihat pada Gambar 5.4. berikut ini



Gambar 5.4. Evaluasi Pendeteksian

Kegagalan pendeteksian *real-time* tersebut dimungkinkan karena *error* pembacaan *frame*.

Hasil pendeteksian objek pada kondisi 2 didapatkan hasil seperti pada Tabel 5.5.

Tabel 5.5. Hasil Pendeteksian Objek Diam Kondisi 2

No.	Objek	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
			Tipe 1	Tipe 2	Tipe 3	
1	Merah	991	9	0	0	99,1 %
2	Hijau	994	6	0	0	99,4 %
3	Ungu	978	22	0	0	97,8 %

Kemampuan pendeteksian tertinggi dimiliki objek hijau dengan kemampuan pendeteksian sebesar 99,4 % diikuti objek merah dengan 99,1 %, dan terendah dimiliki objek ungu dengan kemampuan pendeteksian sebesar 97,8 %. Kesemua objek tidak terdeteksi adalah objek tidak terdeteksi tipe 1. Hasil pada kondisi 2 merupakan kebalikan dari hasil pendeteksian objek diam pada kondisi 1, hal ini dimungkinkan karena objek dengan warna tersebut menghasilkan perbedaan dalam menanggapi kondisi pendeteksian dimana objek hijau memiliki kecenderungan dengan kemudahan terdeteksi pada kondisi 2 dikarenakan *range* warna yang digunakan lebih banyak yang berada di batas atas.

### 5.2.2. Pendeteksian Objek Merah

#### 5.2.2.1. Pendeteksian Objek Merah Kondisi 1

Pendeteksian objek merah yang dilakukan dengan kondisi 1 didapatkan hasil seperti Tabel 5.6. Tabel tersebut menunjukkan tingkat kemampuan pendeteksian semakin kecil seiring dengan besarnya  $\theta_0$  (sudut pembetulan gerakan awal).



Tabel 5.6. Hasil Pendeteksian Objek Merah pada Kondisi 1

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	38	2	0	0	<b>95 %</b>
2	10	40	38	2	0	0	<b>95 %</b>
3	15	40	38	2	0	0	<b>95 %</b>
4	20	40	40	0	0	0	<b>100 %</b>
5	25	40	40	0	0	0	<b>100 %</b>
6	30	40	35	5	0	0	<b>87,5 %</b>
7	35	40	33	7	0	0	<b>82,5 %</b>
8	40	40	34	6	0	0	<b>85 %</b>
9	45	40	26	14	0	0	<b>65 %</b>
10	50	40	26	14	0	0	<b>65 %</b>
11	55	40	29	11	0	0	<b>72,5 %</b>
12	60	40	34	6	0	0	<b>85 %</b>

Kemampuan pendeteksian tertinggi diperoleh ketika  $\theta_0 = 20^\circ$  dan  $25^\circ$  dengan kemampuan pendeteksian sebesar 100%, sedangkan kemampuan pendeteksian terendah terjadi ketika  $\theta_0 = 45^\circ$  dan  $50^\circ$  dengan kemampuan pendeteksian sebesar 65%. Tetapi ada kenaikan kemampuan pendeteksian yang signifikan pada  $\theta_0 = 55^\circ$  dan  $\theta_0 = 60^\circ$ . Kenaikan pada saat  $\theta_0 = 55^\circ$  dan  $\theta_0 = 60^\circ$  dimana kemampuan pendeteksian yang diperoleh dari pendeteksian sebesar 72,5% dan 85% padahal kemampuan pendeteksian sebelumnya pada  $\theta_0 = 45^\circ$  besar dari kemampuan pendeteksian tersebut adalah 65%. Kenaikan kemampuan pendeteksian pada saat  $\theta_0 = 55^\circ$  dan  $\theta_0 = 60^\circ$  lebih disebabkan *frame* yang didapatkan pada saat *real-time* pendeteksian objek memiliki bentuk lingkaran yang masih bagus.



Hasil evaluasi *frame* yang tidak dapat mendeteksi keberadaan objek, diperoleh hasil seperti terlihat pada Tabel 5.7.

Tabel 5.7. Hasil Evaluasi Objek Merah Tidak Terdeteksi

No.	0 <sub>0</sub>	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	2	2	0	0	0	-
2	10	2	2	0	0	0	-
3	15	2	2	0	0	0	-
4	20	0	0	0	0	0	-
5	25	0	0	0	0	0	-
6	30	5	4	1	0	0	4/1
7	35	7	4	3	0	0	4/3
8	40	6	4	2	0	0	4/2
9	45	14	4	10	0	0	4/10
10	50	14	5	9	0	0	5/9
11	55	11	3	8	0	0	3/8
12	60	6	3	3	0	0	3/3

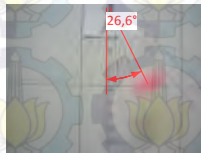
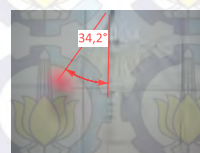
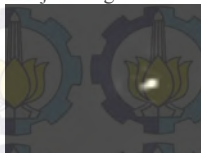
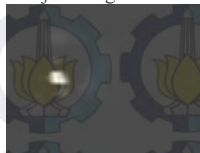
Evaluasi pada  $\theta \leq 25^\circ$  didapatkan objek yang sebelumnya tidak terdeteksi menunjukkan objek tersebut bisa dideteksi. Jika dilihat dari kecepatan objek, kecepatan objek maksimum yang didapatkan terletak pada  $\theta_0 = 25^\circ$  dengan keberadaan objek pada  $\theta = 1^\circ$ , pada saat tersebut kecepatan objek adalah 0,566633 m/s. *Frame* yang dihasilkan terlihat seperti pada Gambar 5.5.



a. Citra Hasil Pendeteksian

b. Citra *Threshold* PendeteksianGambar 5.5. Objek  $\theta_o = 25^\circ$  dengan  $\theta_r = 1^\circ$ 

Seperti terlihat pada Gambar 5.5. tersebut, citra *threshold* yang terbentuk masih terlihat bagus. Sedangkan untuk nilai  $\theta_o \geq 30^\circ$  didapatkan objek banyak yang tidak dapat dideteksi pada saat bergerak. Hal tersebut seperti terlihat pada Gambar 5.6. Gambar tersebut merupakan tiga buah *frame* pertama yang dihasilkan pada saat  $\theta_o = 60^\circ$ .

a. Objek  $\theta = 59^\circ$ b. Objek Bergerak  $\theta = 27^\circ$ c. Objek Bergerak  $\theta = 34^\circ$ d. Citra *Threshold*  $\theta = 59^\circ$ e. Citra *Threshold*  $\theta = 27^\circ$ f. Citra *Threshold*  $\theta = 34^\circ$ Gambar 5.6. Hasil Pendeteksian Objek Bergerak  $\theta_o$  Sebesar  $60^\circ$ 

Objek bergerak berada pada Gambar 5.6.b dan 5.6.c. dengan besar sudut  $\theta$  yang terbentuk dari citra sebesar  $27^\circ$  dan sebesar  $34^\circ$  dari garis tengah. Citra *threshold* yang didapatkan dari objek bergerak tersebut terlihat seperti pada Gambar 5.6.e dan 5.6.f. Hasil *threshold* tersebut menunjukkan objek bergerak pada

saat posisi  $\theta = 27^\circ$  menghasilkan *threshold* kurang bagus dari pada saat objek berada di posisi  $\theta = 34^\circ$  dan  $\theta = 59^\circ$ , hal ini diakibatkan karena perbedaan kecepatan pada saat objek berada diposisi tersebut. Kecepatan objek pada  $\theta = 27^\circ$ ,  $\theta = 34^\circ$  dan  $\theta = 59^\circ$  yaitu 1,032382 m/s, 0,88118 m/s dan 0,11703 m/s. Semakin besar kecepatan objek pada posisi tersebut, semakin tidak bagus citra *threshold* yang dihasilkan.

### 5.2.2.2. Pendeteksian Objek Merah kondisi 2

Pendeteksian objek merah yang dilakukan dengan kondisi 2 didapatkan hasil; seperti pada Tabel 5.8.

Tabel 5.8. Hasil Pendeteksian Objek Hijau pada Kondisi 2

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	39	1	0	0	97,5 %
2	10	40	39	1	0	0	97,5 %
3	15	40	38	2	0	0	95 %
4	20	40	40	0	0	0	100 %
5	25	40	40	0	0	0	100 %
6	30	40	37	3	0	0	92,5 %
7	35	40	40	0	0	0	100 %
8	40	40	36	4	0	0	90 %
9	45	40	40	0	0	0	100 %
10	50	40	36	4	0	0	90 %
11	55	40	35	5	0	0	87,5 %
12	60	40	37	3	0	0	92,5 %

Tabel 5.8 tersebut menunjukkan kemampuan pendeteksian objek tertinggi berada pada saat  $\theta_0 = 20^\circ$ ,  $25^\circ$ ,  $35^\circ$ , dan  $45^\circ$  dengan tingkat kemampuan pendeteksian sebesar 100%. Sedangkan kemampuan pendeteksian terendah berada pada saat  $\theta_0 = 55^\circ$  yaitu sebesar 87.5%. Walaupun terlihat memiliki perbedaan kemampuan

pendeteksian yang tidak terlampaui jauh, tetapi masih mempunyai kecenderungan semakin mengecilnya kemampuan pendeteksian pendeteksian objek secara *real-time*.

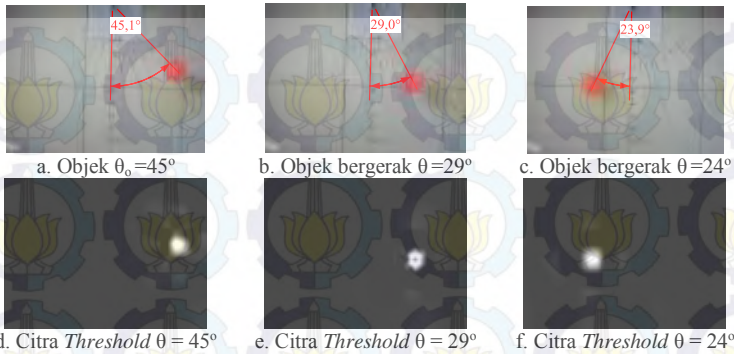
Setelah dilakukan evaluasi terhadap *frame* yang tidak dapat mendeteksi didapatkan hasil seperti terlihat pada Tabel 5.9.

Tabel 5.9. Hasil Evaluasi Objek Merah Tidak Terdeteksi Kondisi 2

No.	0 <sub>0</sub>	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	1	1	0	0	0	-
2	10	1	1	0	0	0	-
3	15	2	2	0	0	0	-
4	20	0	0	0	0	0	-
5	25	0	0	0	0	0	-
6	30	3	3	0	0	0	-
7	35	0	0	0	0	0	-
8	40	4	4	0	0	0	-
9	45	0	0	0	0	0	-
10	50	4	2	2	0	0	2/2
11	55	5	3	2	0	0	3/2
12	60	3	1	2	0	0	1/2

Dari Tabel 5.9. tersebut didapatkan objek yang pada pendeteksian *real-time* dengan  $\theta_0 \leq 45^\circ$ , objek yang semula tidak terdeteksi didapatkan pada saat evaluasi hasil *real-time* tidak didapatkan masalah pendeteksian. Pada saat  $\theta_0 = 45^\circ$  didapatkan tiga buah *frame* lintasan pertama seperti pada Gambar 5.7.

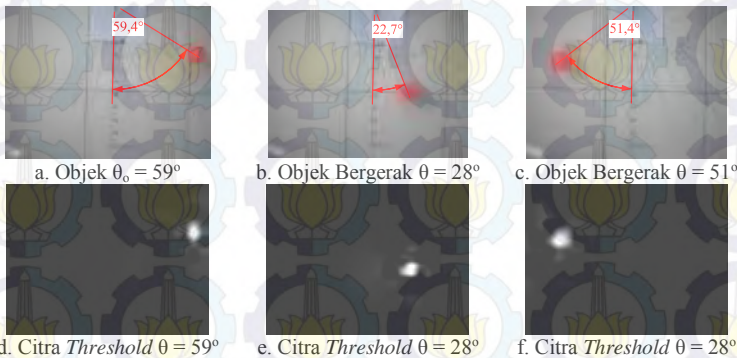




Gambar 5.7. Lintasan Pertama Objek Merah dengan  $\theta_o = 45^\circ$

Pada saat  $\theta_o = 45^\circ$ , kecepatan objek pada saat bergerak dengan posisi objek  $\theta = 29^\circ$  dan  $\theta = 24^\circ$  yaitu 0,6633 m/s dan 0,7691 m/s. Kegagalan pendeteksian *real-time* terjadi ketika  $\theta_o \geq 50^\circ$ , dimana tetap ada objek yang tidak dapat terdeteksi.

Saat  $\theta_o = 60^\circ$  didapatkan lintasan pertama hasil pendeteksian seperti pada Gambar 5.8.



Gambar 5.8. Lintasan Pertama Objek Merah dengan  $\theta_o = 60^\circ$

Gambar 5.8 tersebut menunjukkan pada saat  $\theta_0 = 60^\circ$  objek tersebut tidak terdeteksi, hal tersebut dikarenakan *threshold* yang didapatkan kurang bagus. Sedangkan pada saat bergerak dan menunjukkan  $\theta = 28^\circ$  objek tidak terdeteksi dan pada  $\theta = 51^\circ$  objek terdeteksi. Pada saat objek bergerak dengan  $\theta = 28^\circ$  dan  $\theta = 51^\circ$  objek tersebut memiliki kecepatan sebesar 1,10606 m/s dan 0,41934 m/s. Kecepatan tersebut mempengaruhi citra *threshold* yang dihasilkan.

### 5.2.3. Pendeteksian Objek Hijau

#### 5.2.3.1. Pendeteksian Objek Hijau Kondisi 1

Pendeteksian objek hijau yang dilakukan dengan kondisi 1 didapatkan hasil seperti Tabel 5.10.

Tabel 5.10. Hasil Pendeteksian Objek Hijau pada Kondisi 1

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	38	2	0	0	<u>95 %</u>
2	10	40	40	1	0	0	<u>97,5 %</u>
3	15	40	40	0	0	0	<u>100 %</u>
4	20	40	39	1	0	0	<u>97,5 %</u>
5	25	40	35	4	1	0	<u>87,5 %</u>
6	30	40	30	10	0	0	<u>75 %</u>
7	35	40	32	8	0	0	<u>80 %</u>
8	40	40	30	10	0	0	<u>75 %</u>
9	45	40	28	12	0	0	<u>70 %</u>
10	50	40	30	10	0	0	<u>75 %</u>
11	55	40	26	14	0	0	<u>65 %</u>
12	60	40	23	17	0	0	<u>57,5 %</u>

Kemampuan pendeteksian yang didapatkan dari pendeteksian tersebut cenderung menurun seiring kenaikan sudut  $\theta_0$ . Kemampuan pendeteksian tertinggi didapatkan pada saat

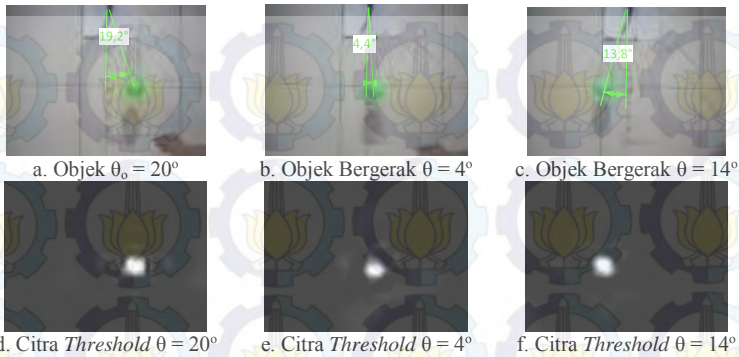
pengujian dengan  $\theta_0 = 15^\circ$  yaitu sebesar 100%, sedangkan untuk kemampuan pendeteksian terendah diperoleh ketika  $\theta_0 = 60^\circ$  dengan kemampuan pendeteksian sebesar 57.5%. Pada saat  $\theta_0 = 60^\circ$  terdapat 17 *frame* yang tidak dapat mendeteksi keberadaan objek.

Setelah dilakukan evaluasi terhadap *frame* yang tidak dapat mendeteksi keberadaan objek didapatkan seperti terlihat pada Tabel 5.11.

Tabel 5.11. Hasil Evaluasi Objek Hijau Tidak Terdeteksi Kondisi 1

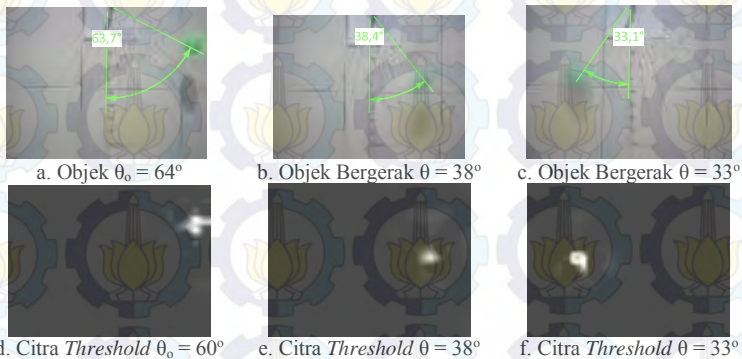
No.	0 <sub>0</sub>	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	2	2	0	0	0	-
2	10	1	1	0	0	0	-
3	15	0	0	0	0	0	-
4	20	3	3	0	0	0	-
5	25	5	3	1	1	0	-
6	30	10	8	2	0	0	8/2
7	35	8	1	7	0	0	1/7
8	40	10	4	6	0	0	4/6
9	45	12	1	11	0	0	1/11
10	50	10	2	8	0	0	2/8
11	55	14	6	8	0	0	6/8
12	60	17	6	11	0	0	6/11

Tabel 5.11 tersebut didapatkan hasil evaluasi pada saat  $\theta_0 \leq 25^\circ$  keberadaan objek masih dapat terdeteksi. Pada saat  $\theta_0 = 20^\circ$  didapatkan *frame* lintasan pertama seperti pada Gambar 5.9. *Frame* mendeteksi keberadaan objek bergerak pada saat objek berada pada posisi  $\theta = 4^\circ$  dan  $\theta = 14^\circ$ . Kecepatan pada posisi tersebut yaitu 0,4444 m/s dan 0,3145 m/s.



Gambar 5.9. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $20^\circ$

Sedangkan pada saat  $\theta_0 \geq 30^\circ$  masih banyak objek yang tidak bisa dideteksi dengan objek tidak terdeteksi terdapat pada 11 *frame* pada percobaan dengan  $\theta_0 = 45^\circ$  dan  $60^\circ$ . Hal tersebut dapat dilihat dari lintasan pertama pendeteksian pada saat  $\theta_0 = 60^\circ$  seperti ditunjukkan pada Gambar 5.10.



Gambar 5.10. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $60^\circ$



Objek bergerak berada pada Gambar 5.10.b dan 5.10.c. dengan besar sudut  $\theta$  yang terbentuk dari citra yang didapatkan tersebut sebesar  $38^\circ$  dan  $33^\circ$  dari garis tengah. Kecepatan objek pada saat posisi tersebut yaitu  $0,783616 \text{ m/s}$  dan  $0,90275 \text{ m/s}$ . Citra *threshold* yang didapatkan terlihat pada Gambar 5.10.e dan 5.10.f. hasil citra *threshold* tersebut tidak bisa membentuk lingkaran sehingga pada saat evaluasi objek tidak terdeteksi tetap tidak dapat dideteksi.

### 5.2.3.2. Pendeteksian Objek Hijau Kondisi 2

Pendeteksian objek hijau yang dilakukan dengan kondisi 2 didapatkan hasil seperti Tabel 5.12.

Tabel 5.12. Hasil Pendeteksian Objek Hijau pada Kondisi 2

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	40	0	0	0	<u>100 %</u>
2	10	40	40	0	0	0	<u>100 %</u>
3	15	40	40	0	0	0	<u>100 %</u>
4	20	40	39	1	0	0	<u>97,5 %</u>
5	25	40	40	0	0	0	<u>100 %</u>
6	30	40	39	1	0	0	<u>97,5 %</u>
7	35	40	37	3	0	0	<u>92,5 %</u>
8	40	40	38	2	0	0	<u>95 %</u>
9	45	40	39	1	0	0	<u>97,5 %</u>
10	50	40	37	3	0	0	<u>92,5 %</u>
11	55	40	36	4	0	0	<u>90 %</u>
12	60	40	35	5	0	0	<u>87,5 %</u>

Kemampuan pendeteksian secara *real-time* menunjukkan kemampuan pendeteksian tertinggi diperoleh ketika  $\theta_0 = 5^\circ, 10^\circ, 15^\circ$ , dan  $25^\circ$  sebesar 100%. Sedangkan kemampuan pendeteksian terendah diperoleh ketika  $\theta_0 = 60^\circ$  dengan tingkat kemampuan

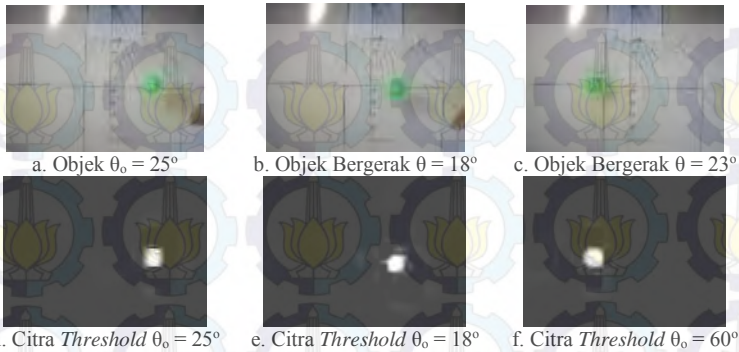
pendeteksian sebesar 87.5%. Kemampuan pendeteksian secara *real-time* cenderung menurun seiring dengan kenaikan  $\theta_0$ .

Setelah dilakukan evaluasi terhadap *frame* yang tidak dapat mendeteksi keberadaan objek secara *real-time*, didapatkan hasil seperti pada Tabel 5.13.

Tabel 5.13. Hasil Evaluasi Objek Hijau Tidak Terdeteksi Kondisi 2

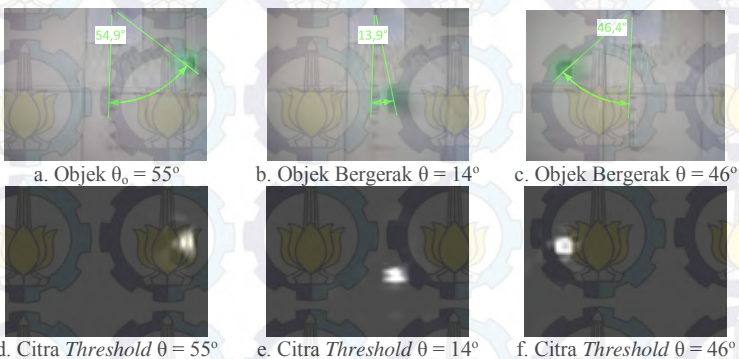
No.	θ <sub>0</sub>	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	0	0	0	0	0	-
2	10	0	0	0	0	0	-
3	15	0	0	0	0	0	-
4	20	1	1	0	0	0	1/0
5	25	0	0	0	0	0	-
6	30	1	1	0	0	0	1/0
7	35	3	2	1	0	0	2/1
8	40	2	1	1	0	0	1/1
9	45	1	1	0	0	0	1/0
10	50	3	1	2	0	0	2/1
11	55	4	0	4	0	0	0/4
12	60	5	3	2	0	0	3/2

Pada saat  $\theta_0 = 25^\circ$  didapatkan tiga buah *frame* lintasan pertama seperti pada Gambar 5.11. *frame* tersebut menunjukkan objek bergerak dengan  $\theta = 18^\circ$  dan  $\theta = 23^\circ$ . Kecepatan pada saat posisi tersebut yaitu 0,549 m/s dan 0,2032 m/s.



Gambar 5.11. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $25^\circ$

Hasil evaluasi menunjukkan *frame* yang tidak dapat terdeteksi terbanyak diperoleh ketika percobaan dengan  $\theta_0 = 55^\circ$  dengan 4 *frame* tidak terdeteksi keberadaan objek. Gambar 5.12 merupakan citra hasil percobaan dengan  $\theta_0 = 55^\circ$  yang menunjukkan tiga *frame* lintasan pertama hasil pendeteksian yaitu objek bergerak dengan  $\theta = 14^\circ$  dan  $\theta = 46^\circ$ .



Gambar 5.12. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $55^\circ$

Kecepatan objek bergerak pada saat dengan  $\theta = 14^\circ$  dan  $\theta = 46^\circ$  yaitu 1,132435 m/s dan 0,447898 m/s.

#### 5.2.4. Pendeteksian Objek Ungu

##### 5.2.4.1. Pendeteksian Objek Ungu Kondisi 1

Pendeteksian objek hijau yang dilakukan dengan kondisi 1 didapatkan hasil seperti Tabel 5.14.

Tabel 5.14. Hasil Pendeteksian Objek Ungu pada Kondisi 1

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	39	1	0	0	97,5 %
2	10	40	40	0	0	0	100 %
3	15	40	40	0	0	0	100 %
4	20	40	40	0	0	0	100 %
5	25	40	37	4	0	0	90 %
6	30	40	36	4	0	0	90 %
7	35	40	33	7	0	0	82,5 %
8	40	40	28	12	0	0	75 %
9	45	40	32	8	0	0	75 %
10	50	40	31	9	0	0	72,5 %
11	55	40	26	14	0	0	65 %
12	60	40	25	15	0	0	62,5 %

Tabel 5.14 menunjukkan kecenderungan menurunnya tingkat kemampuan pendeteksian *real-time* seiring bertambahnya  $\theta_0$ . Kemampuan pendeteksian tertinggi diperoleh ketika percobaan dengan  $\theta_0 = 10^\circ$ ,  $15^\circ$ , dan  $20^\circ$  sebesar 100%. Sedangkan tingkat kemampuan pendeteksian terendah diperoleh dari percobaan dengan  $\theta_0 = 10^\circ$  dengan akurasi pendeteksian *real-time* sebesar 62.5% dengan 15 *frame* dari 40 *frame* tidak berhasil mendeteksi keberadaan objek.

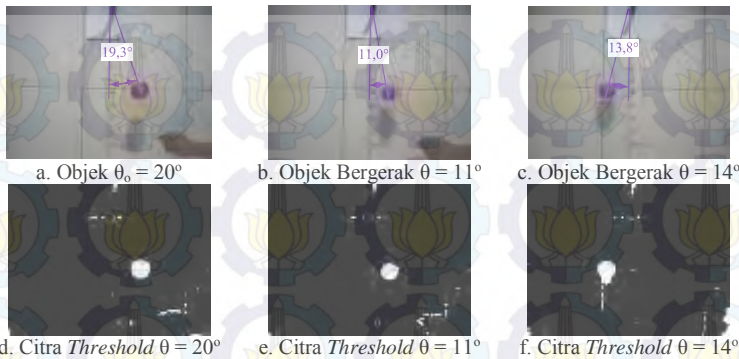


Setelah dilakukan evaluasi terhadap *frame* yang tidak dapat mendeteksi keberadaan objek pada saat *real-time* didapatkan hasil seperti terlihat pada Tabel 5.11.

Tabel 5.15. Hasil Evaluasi Objek Ungu Tidak Terdeteksi Kondisi 1

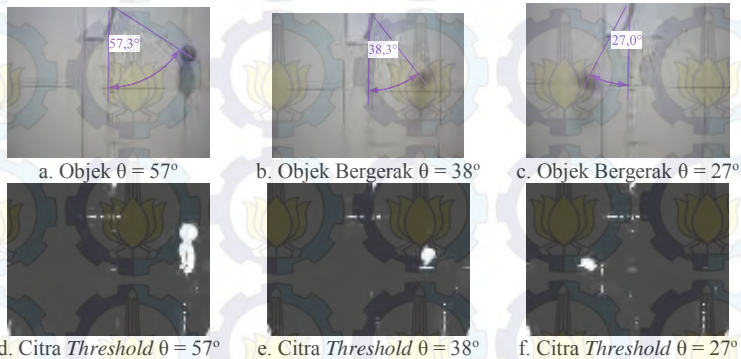
No.	0 <sub>o</sub>	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksiian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	1	1	0	0	0	1/0
2	10	0	0	0	0	0	-
3	15	0	0	0	0	0	-
4	20	0	0	0	0	0	-
5	25	4	4	0	0	0	4/0
6	30	4	4	0	0	0	4/0
7	35	7	3	4	0	0	3/4
8	40	12	6	6	0	0	6/6
9	45	8	3	5	0	0	3/5
10	50	4	4	5	0	0	4/5
11	55	14	7	7	0	0	7/7
12	60	15	7	8	0	0	7/8

Tabel 5.15 menunjukkan ketika  $\theta_0 \leq 30^\circ$ , objek yang tidak terdeteksi keberadaanya pada saat *real-time* seharusnya masih dapat dideteksi. Lintasan pertama pada saat  $\theta_0 = 20^\circ$  didapatkan objek bergerak dengan posisi  $\theta = 11^\circ$  dan  $\theta = 14^\circ$  seperti ditunjukkan Gambar 5.13. kecepatan objek tersebut yaitu 0,37248 m/s dan 0,314524 m/s.



Gambar 5.13. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $20^\circ$

Gambar 5.14 menunjukkan citra hasil pendeteksian dan citra *threshold* dari lintasan pertama percobaan dengan  $\theta_0 = 55^\circ$ .



Gambar 5.14. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $55^\circ$

Objek bergerak berada pada Gambar 5.14.b dan 5.14.c. dengan besar sudut  $\theta$  yang terbentuk dari citra yang didapatkan tersebut sebesar  $38^\circ$  dan sebesar  $27^\circ$ . kecepatan objek bergerak pada posisi tersebut yaitu  $0,676155 \text{ m/s}$  dan  $0,930192 \text{ m/s}$ .

#### 5.2.4.2. Pendeteksian Objek Ungu Kondisi 2

Pendeteksian objek hijau yang dilakukan dengan kondisi 2 didapatkan hasil seperti Tabel 5.16.

Tabel 5.16. Hasil Pendeteksian Objek Ungu pada Kondisi 2

No.	$\theta_0$	Jumlah <i>frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek			Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	
1	5	40	40	0	0	0	100 %
2	10	40	40	0	0	0	100 %
3	15	40	39	1	0	0	97,5 %
4	20	40	39	1	0	0	97,5 %
5	25	40	39	1	0	0	97,5 %
6	30	40	39	1	0	0	97,5 %
7	35	40	38	2	0	0	95 %
8	40	40	38	2	0	0	95 %
9	45	40	37	3	0	0	92,5 %
10	50	40	38	2	0	0	95 %
11	55	40	35	5	0	0	87,5 %
12	60	40	35	5	0	0	87,5 %

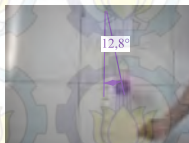
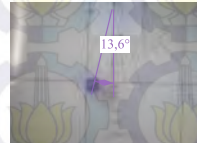
Tabel 5.16 menunjukkan kecenderungan menurunnya kemampuan pendeteksian *real-time* seiring dengan bertambahnya besar  $\theta_0$ . Kemampuan pendeteksian tertinggi diperoleh ketika  $\theta_0 = 5^\circ$  dan  $10^\circ$  dengan kemampuan pendeteksian sebesar 100%. Sedangkan kemampuan pendeteksian terendah didapatkan pada saat percobaan dengan  $\theta_0 = 55^\circ$  dan  $60^\circ$  dengan kemampuan pendeteksian yang didapat sebesar 87.5% dimana terdapat 5 *frame* yang tidak dapat mendeteksi keberadaan objek secara *real-time*.

Setelah dilakukan evaluasi terhadap *frame* yang tidak dapat mendeteksi keberadaan objek secara *real-time* didapatkan hasil seperti terlihat pada Tabel 5.17.

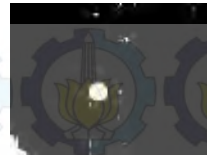
Tabel 5.17. Hasil Evaluasi Objek Ungu Tidak Terdeteksi Kondisi 2

No.	$\theta_0$	Jumlah <i>Frame</i>	Hasil Evaluasi				Perbandingan Terdeteksi/ Tidak Terdeteksi
			Objek Terdeteksi	Kesalahan Pendeteksian Objek			
				Tipe 1	Tipe 2	Tipe 3	
1	5	0	0	0	0	0	-
2	10	0	0	0	0	0	-
3	15	1	1	0	0	0	1/0
4	20	1	1	0	0	0	1/0
5	25	1	1	0	0	0	1/0
6	30	1	1	0	0	0	1/0
7	35	2	2	0	0	0	2/0
8	40	2	2	0	0	0	2/0
9	45	3	2	1	0	0	2/1
10	50	2	2	0	0	0	2/0
11	55	5	3	2	0	0	3/2
12	60	5	3	2	0	0	3/2

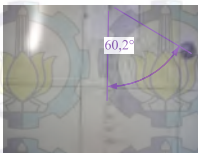
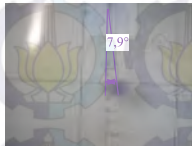
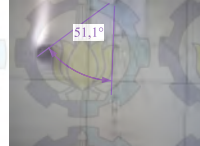
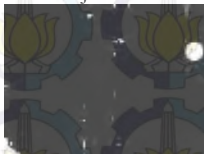
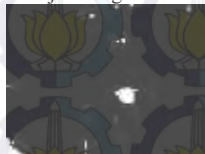
*Frame* yang didapat ketika  $\theta_0 \leq 40^\circ$  dari pendeteksian *real-time* ternyata seharusnya masih bisa dideteksi. Lintasan pertama pendeteksian objek dengan  $\theta_0 = 13^\circ$  seperti terlihat pada Gambar 5.15. Dari lintasan tersebut didapatkan objek bergerak pada posisi  $\theta = 13^\circ$  dan  $\theta = 1^\circ$ . Pada posisi tersebut objek mempunyai kecepatan 0 m/s dan 0,227214 m/s. Citra *threshold* yang didapatkan masih terlihat bagus dan utuh.

a. Objek  $\theta = 13^\circ$ b. Objek Bergerak  $\theta = 1^\circ$ c. Objek Bergerak  $\theta = 13^\circ$ Gambar 5.15. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $13^\circ$



d. Citra *Threshold*  $\theta = 13^\circ$ e. Citra *Threshold*  $\theta = 1^\circ$ f. Citra *Threshold*  $\theta = 13^\circ$ Gambar 5.15. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $13^\circ$  (Lanjutan)

Sedangkan untuk  $\theta_0 = 60^\circ$  lintasan pertama objek bergerak didapatkan pada  $\theta = 8^\circ$  dan  $\theta = 51^\circ$  seperti ditunjukkan pada Gambar 5.16. pada posisi tersebut objek bergerak dengan kecepatan 1,2848 m/s dan 0,4186 m/s.

a. Objek  $\theta = 60^\circ$ b. Objek Bergerak  $\theta = 8^\circ$ c. Objek Bergerak  $\theta = 51^\circ$ d. Citra *Threshold*  $\theta = 60^\circ$ e. Citra *Threshold*  $\theta = 8^\circ$ f. Citra *Threshold*  $\theta = 51^\circ$ Gambar 5.16. Hasil Pendeteksian Objek Bergerak  $\theta_0$  Sebesar  $60^\circ$ 

### 5.2.5. Perbandingan Hasil Pendeteksian *Real-Time*

Perbandingan hasil kemampuan pendeteksian secara *real-time* semua kondisi terlihat pada Tabel 5.18. sedangkan untuk hasil evaluasi pendeteksian terlihat pada Tabel 5.19.

Tabel 5.18. Perbandingan Kemampuan Pendeteksian *Real-Time*.

No.	$\theta_0$	Kemampuan Pendeteksian (%)					
		Kondisi 1			Kondisi 2		
		Merah	Hijau	Ungu	Merah	Hijau	Ungu
1	5	95 %	95 %	97,5 %	97,5 %	100 %	100 %
2	10	95 %	97,5 %	100 %	97,5 %	100 %	100 %
3	15	95 %	100 %	100 %	95 %	100 %	97,5 %
4	20	100 %	97,5 %	100 %	100 %	97,5 %	97,5 %
5	25	100 %	87,5 %	90 %	100 %	100 %	97,5 %
6	30	87,5 %	75 %	90 %	92,5 %	97,5 %	97,5 %
7	35	82,5 %	80 %	82,5 %	100 %	92,5 %	95 %
8	40	85 %	75 %	75 %	90 %	95 %	95 %
9	45	65 %	70 %	75 %	100 %	97,5 %	92,5 %
10	50	65 %	75 %	72,5 %	90 %	92,5 %	95 %
11	55	72,5 %	65 %	65 %	87,5 %	90 %	87,5 %
12	60	85 %	57,5 %	62,5 %	92,5 %	87,5 %	87,5 %

Secara umum kemampuan pendeteksian dari pendeteksian *real-time* yaitu semakin besar  $\theta_0$  maka kemampuan pendeteksian yang diperoleh akan semakin kecil. Hal ini disebabkan karena semakin besar  $\theta_0$  maka banyak dari *frame* yang dijadikan sebagai data kemampuan pendeteksian memiliki kecepatan lebih tinggi dari  $\theta_0$  yang lebih kecil. Dari Tabel 5.18 tersebut diketahui bahwa pendeteksian pada kondisi 2 memiliki kemampuan pendeteksian yang lebih bagus dibandingkan pada kondisi 1. Hal tersebut juga terlihat pada saat evaluasi pendeteksian real-time pada Tabel 5. 19. Hal tersebut terjadi dikarenakan *range* warna yang dijadikan pembentuk citra *threshold* hanya dilakukan pada salah satu kondisi saja.

Tabel 5.19. Hasil Evaluasi Pendeteksian *Real-Time*

No.	$\theta_0$	Evaluasi (Terdeteksi/Tidak Terdeteksi)					
		Kondisi 1			Kondisi 2		
		Merah	Hijau	Ungu	Merah	Hijau	Ungu
1	5	-	-	1/0	-	-	-
2	10	-	-	-	-	-	-

3	15	-	-	-	-	-	1/0
4	20	-	-	-	-	1/0	1/0
5	25	-	-	-	-	-	1/0
6	30	4/1	8/2	4/0	-	1/0	1/0
7	35	4/3	1/7	3/4	-	2/1	2/0
8	40	4/2	4/6	6/6	-	1/1	2/0
9	45	4/10	1/11	3/5	-	1/0	2/1
10	50	5/9	2/8	4/5	2/2	2/1	2/0
11	55	3/8	6/8	7/7	3/2	0/4	3/2
12	60	3/3	6/11	7/8	1/2	3/2	3/2

### 5.3. Multi Objek

Pendeteksian multi objek dilakukan pada bola dengan warna dominan merah, hijau, dan biru. Pendeteksian dilakukan pada objek tidak bergerak (diam) dan objek bergerak.

#### 5.3.1. Pendeteksian Multi Objek Diam

##### 5.3.1.1. Pendeteksian Multi Objek Diam Kondisi 1

Pendeteksian multi objek dalam keadaan tidak bergerak yang dilakukan dengan kondisi 1 didapatkan hasil seperti Tabel 5.20.

Tabel 5.20. Hasil Pendeteksian Multi Objek tidak Bergerak Kondisi 1

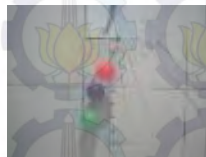
No.	Objek	Jumlah <i>Frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek				Kemamp uan pendetek sian (%)
				Tipe 1	Tipe 2	Tipe 3	Total	
1	Merah	100	100	0	0	0	12	<b>88 %</b>
2	Hijau		98	2	0	0		
3	Ungu		90	10	0	0		

Kemampuan pendeteksian yang didapatkan dari pendeteksian objek tersebut yaitu sebesar 88 % dengan jumlah objek yang tidak bisa dideteksi sebanyak 12 dengan rincian tidak bisa mendeteksi keberadaan objek hijau sebanyak 2 buah *frame* dan keberadaan

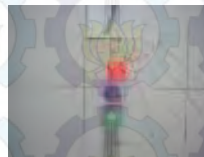


objek ungu sebanyak 10 *frame*. Objek yang tidak bisa dideteksi tersebut yaitu tidak dapat mendeteksi keberadaan objek hijau pada *frame* ke-4 dan ke-10. Sedangkan yang tidak bisa mendeteksi keberadaan objek ungu pada *frame* ke-11, 27, 35, 36, 43, 46, 47, 67, 80, dan *frame* ke-99.

Gambar 5.17 menunjukkan hasil citra pendeteksian dan evaluasi dari *frame* ke-1 dan ke-4 yang terdapat sebagian objek yang tidak dapat dideteksi.



a. Citra Asli *Frame* ke-1



b. Hasil Pendeteksian *Frame* ke-4



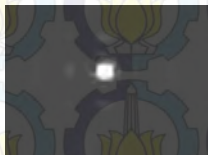
c. Hasil Evaluasi *Frame* ke-1



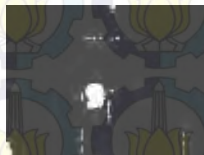
d. Hasil Evaluasi *Frame* ke-4

Gambar 5.17. Citra Hasil Pendeteksian dan Evaluasi *Frame* ke-1 dan ke-4

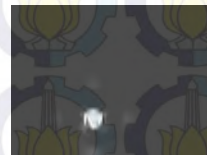
Gambar 5.18 menunjukkan hasil citra *threshold* yang terbentuk dari *frame* ke-1 dan *frame* ke-4 terlihat memiliki bentuk yang biner yang bagus.



a. Citra *Threshold* Merah  
*Frame* ke-1



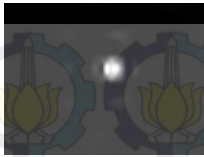
b. Citra *Threshold* Ungu  
*Frame* ke-1



c. Citra *Threshold* Hijau  
*Frame* ke-1

Gambar 5.18. Citra *Threshold* Hasil Evaluasi *Frame* ke-1 dan ke-4

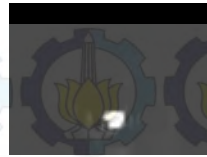




d. Citra *Threshold* Merah  
Frame ke-4



e. Citra *Threshold* Ungu  
Frame ke-4



f. Citra *Threshold* Hijau  
Frame ke-4

Gambar 5.18. Citra *Threshold* Hasil Evaluasi *Frame* ke-1 dan ke-4 (Lanjutan)

Hasil evaluasi keseluruhan *frame* yang tidak dapat dideteksi secara *real-time* didapatkan hasil seperti pada Tabel 5.21.

Tabel 5.21. Hasil Evaluasi Objek Tidak Terdeteksi

No.	Objek	Kesalahan Pendeteksian Objek			Hasil Evaluasi	
		Tipe 1	Tipe 2	Tipe 3	Terdeteksi	Tidak Terdeteksi
1	Merah	0	0	0	0	0
2	Hijau	2	0	0	2	0
3	Ungu	10	0	0	9	1

Tabel 5.21 tersebut menunjukkan pada saat evaluasi objek tidak terdeteksi terdapat 1 buah *frame* yang tidak lolos evaluasi. Objek tersebut merupakan objek ungu yang berada pada *frame* ke-46. Hal tersebut disebabkan karena *threshold* objek ungu yang dihasilkan tidak memenuhi syarat pendeteksian lingkaran. Citra *threshold* dari *frame* ke-46 seperti terlihat pada Gambar 5.19.



Gambar 5.19. Citra *Threshold* Hasil Evaluasi *Frame* ke-46

### 5.3.1.2. Pendeteksian Multi Objek Diam Kondisi 2

Pendeteksian multi objek dalam keadaan diam yang dilakukan pada kondisi 2 didapatkan hasil seperti Tabel 5.22.

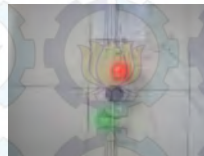
Tabel 5.22. Hasil Pendeteksian Multi Objek tidak Bergerak Kondisi 2

No.	Objek	Jumlah <i>Frame</i>	Objek Terdeteksi	Kesalahan Pendeteksian Objek				Kemampuan Pendeteksian (%)
				Tipe 1	Tipe 2	Tipe 3	Total	
1	Merah	100	100	0	0	0	2	98 %
2	Hijau		100	0	0	0		
3	Ungu		98	2	0	0		

Dari Tabel data yang diperoleh, terdapat 2 buah *frame* yang tidak mendeteksi keberadaan objek ungu. *Frame* yang tidak dapat mendeteksi keberadaan objek tersebut yaitu *frame* ke-8 dan 94 seperti terlihat pada Gambar 5.20.



a. Hasil Pendeteksian *Frame* ke-1



b. Hasil Pendeteksian *Frame* ke-94



c. Hasil Evaluasi *Frame* ke-8



d. Hasil Evaluasi *Frame* ke-94

Gambar 5.20. Citra Hasil Pendeteksian dan Evaluasi *Frame* ke-8 dan ke-94

Hasil evaluasi dari kedua *frame* tersebut, didapatkan citra *threshold* seperti pada Gambar 5.21.



Gambar 5.21. Citra *Threshold* Hasil Evaluasi *Frame* ke-8 dan ke-94

Hasil evaluasi *frame* ke-8 dan ke-94 menunjukkan bahwa *frame* tersebut seharusnya bisa dideteksi. Seperti yang terlihat pada Gambar 5.20.b dan 5.20.e citra *threshold* terlihat dapat membentuk lingkaran. Kemungkinan tidak terdeteksinya keberadaan objek ungu pada *frame* ke-8 dan 94 karena program tidak bisa atau melewati proses looping pendeteksian objek ungu.

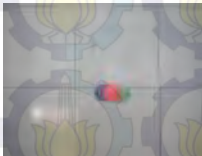
### 5.3.2. Pendeteksian Multi Objek Bergerak

Pendeteksian multi objek bergerak terdapat 3 buah objek yaitu objek merah, hijau, dan ungu. Dari pendeteksian secara *real-time* didapatkan hasil seperti pada Tabel 5.23.

Tabel 5.23. Hasil *Real-time* Pendeteksian Multi Objek Bergerak

No.	Objek	Jumlah <i>Frame</i>	Kesalahan Pendeteksian Objek				Kemampuan Pendeteksian (%)
			Tipe 1	Tipe 2	Tipe 3	Total	
1	Merah	100	-	0	0	-	-
2	Hijau		-	0	0		
3	Ungu		-	0	0		

Kemampuan pendeteksian pada saat pendeteksian *real-time* belum bisa didapatkan. Tabel 5.23. menunjukkan objek yang berada pada tipe 1 objek tidak terdeteksi belum bisa diketahui, tetapi tipe 2 dan tipe 3 objek tidak terdeteksi adalah 0 (tidak ada). Multi objek yang sedang bergerak seperti pada Gambar 5.22 ada sebagian yang saling menutupi.



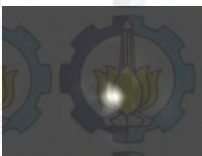
a. Citra Asli Pendeteksian *Real-time*



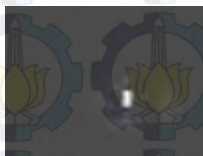
b. Hasil Pendeteksian *Real-time*

Gambar 5.22. Pendeteksian Multi Objek Bergerak

Gambar 5.22 tersebut menghasilkan citra *threshold* seperti terlihat pada Gambar 5.23 pada citra *threshold* tersebut menunjukkan masing-masing hasil *threshold* dari multi objek salah satu *frame* yang sedang bergerak. Ada yang lingkaran penuh ada juga yang setengah lingkaran, hal inilah yang menjadikan kemampuan pendeteksian dari pendeteksian *real-time* tidak bisa ditentukan hanya dengan pengamatan visual. Untuk mendekati kemampuan pendeteksian *real-time* akan dilakukan dengan mengevaluasi seluruh *frame real-time*.



a. Citra *Threshold* Merah



b. Citra *Threshold* Hijau



c. Citra *Threshold* Ungu

Gambar 5.23. Citra *Threshold* Multi Objek



Hasil evaluasi pendeteksian multi objek didapatkan data yaitu dari 100 *frame* tersebut terdapat 4 tambahan pendeteksian multi objek. *Frame* tersebut yaitu *frame* ke-1, 9, 14, dan 29. Gambar 5.24 menunjukkan perbedaan hasil pendeteksian secara *real-time* dan evaluasi pada *frame* ke-1.



a. Pendeteksian *Real-time* *Frame* ke-1



b. Pendeteksian Evaluasi *Frame* ke-1

Gambar 5.24. Perbedaan Hasil Pendeteksian *Real-time* dan Evaluasi

Jika hasil dari pendeteksian evaluasi disamakan dengan kemampuan pendeteksian sebesar 100%, maka kemampuan pendeteksian hasil pendeteksian *real-time* sebesar 96%. Hal ini diakibatkan karena ada objek yang tidak bisa terdeteksi secara *real-time* tetapi dapat terdeteksi pada saat evaluasi pendeteksian objek.

#### 5.4. Pendeteksian Jarak Terjauh Objek Terdeteksi

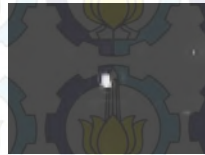
Pencarian jarak terjauh yang dapat dideteksi dilakukan pada objek bola warna hijau dengan bahan sebanyak 40 *frame* seperti terlihat pada Lampiran 5.1 Dari hasil tersebut (Lampiran 5.1), didapatkan jarak terjauh yang dapat dideteksi yaitu objek dengan radius terdeteksi sebesar 20 piksel. Radius tersebut masih di atas radius minimum pendeteksian lingkaran yaitu sebesar 10 piksel. Jarak pendeteksian sebenarnya yaitu berada pada jarak 210 cm. Jarak tersebut merupakan jarak antara kamera dengan objek berada.

Citra asli pendeteksian dan *threshold* yang menunjukkan pendeteksian objek dengan jarak terjauh tersebut terlihat seperti

pada Gambar 5.25. Dari gambar tersebut, diketahui bahwa hasil *threshold* objek masih terlihat bagus.



a. Citra Pendeteksian



b. Citra *Threshold*

Gambar 5.25. Citra Pendeteksian



## BAB IV PENUTUP

Pada bab ini akan dijelaskan mengenai kesimpulan dari hasil uji coba dan evaluasi yang telah dilakukan. Selain itu, terdapat beberapa saran guna penyempurnaan sistem untuk kedepannya.

### 6.1. Kesimpulan

Kesimpulan yang diperoleh dari laporan tugas akhir ini adalah sebagai berikut

1. *Range* RGB yang digunakan dalam pendeteksian yaitu objek merah (160-255, 0-75, 0-90), objek hijau (40-90, 100-255, 60-180), dan objek ungu (30-125, 30-70, 50-150).
2. Hasil pendeteksian secara *real-time* keadaan diam tidak ada yang mencapai 100% baik dilakukan pada kondisi 1 maupun pada kondisi 2. Dari 1000 *frame* inputan akurasi pada kondisi 1 objek merah sebesar 95,7 %, objek hijau 99,6 %, dan objek ungu 99,6 %, sedangkan pada kondisi 2 diperoleh akurasi pendeteksian pada objek merah sebesar 99,4 %, objek hijau 99,1 % dan objek ungu sebesar 97,8 %.
3. Kemampuan pendeteksian pada kondisi 2 mengalami perbedaan dengan pendeteksian kondisi 1, hal ini dikarenakan acuan *range* warna yang digunakan adalah *range* warna dari objek pada kondisi 2. Kemampuan pendeteksian mengalami kenaikan 0,06% pada objek merah dan 0,18% pada objek ungu, sedangkan mengalami penurunan kemampuan pendeteksian pada objek hijau sebesar 0,37%.



4. Objek terjauh yang berhasil dideteksi secara *real-time* yaitu objek dengan radius sebesar 20 piksel atau dengan objek dengan diameter 7 cm dapat terdeteksi pada jarak 210 cm.
5. Hasil pendeteksian evaluasi lebih bagus dibandingkan hasil pendeteksian secara *real-time*, hal ini dikarenakan pendeteksian evaluasi objek bisa menghindari gangguan keterbatasan alat pendeteksian dalam hal ini kamera Usb.

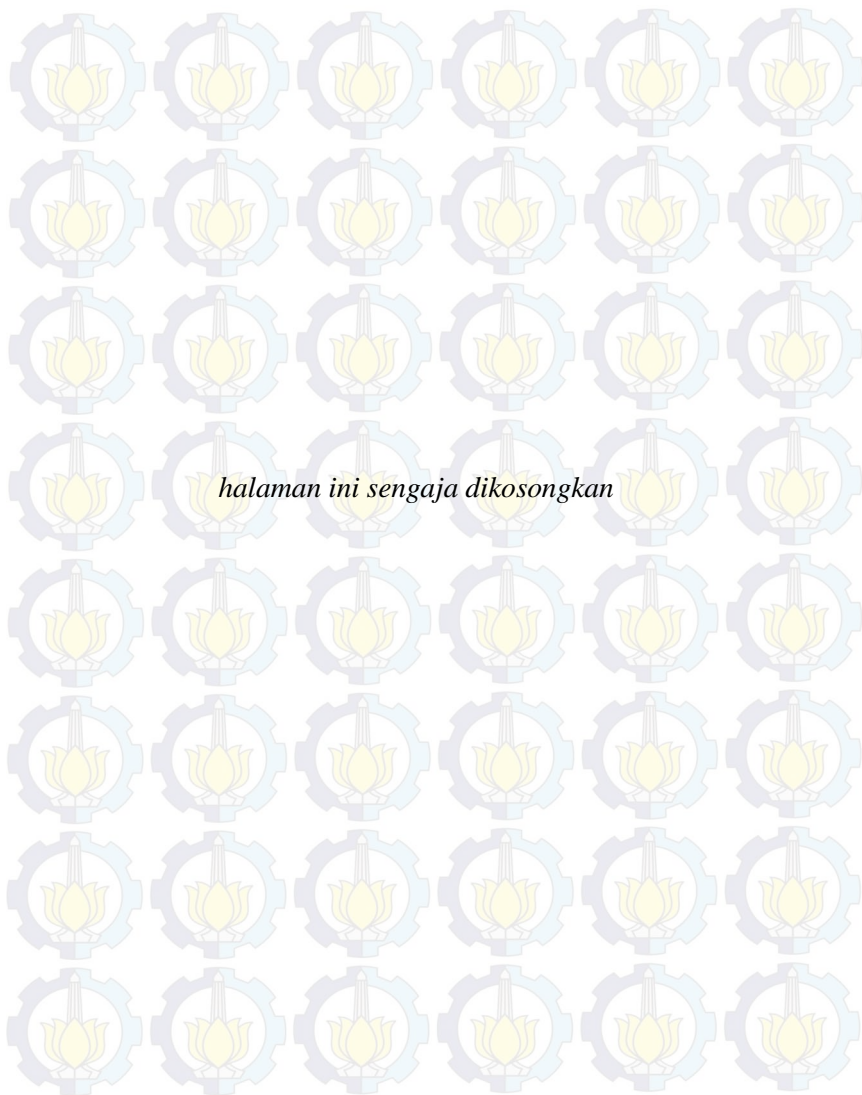
## **6.2. Saran**

Saran pada subbab ini ditujukan untuk pengembangan sistem dengan melakukan penelitian lebih lanjut. Berikut adalah saran yang diajukan.

1. Sistem ini dapat diterapkan pada aplikasi robot yang berjalan secara otomatis.
2. Untuk menunjang jalannya sistem yang lebih baik disarankan sistem dapat membedakan lebih banyak objek.

## DAFTAR PUSTAKA

- [1] Bimbo, D dan Dini, F. (2011), “Particle Filter-Based Visual Tracking With a First Order Dynamic Model and Uncertainty Adaptation”, dalam *Computer Vision and Image Understanding*, Vol. 115, No. 6. Hal. 771–786.
- [2] Yi, Shuangyan dkk. (2014), *Single Object Tracking via Robust Combination of Particle Filter and Sparse Representation*, Signal Processing, Elsevier B.V.
- [3] Yang, Changjiang dkk. (2005), *Fast Multiple Object Tracking via a Hierarchical Particle Filter*, Department of Computer Science, University of Maryland, USA.
- [4] Gonzalez, C. Rafael, and Woods, E. Richard, (2008), *Digital Image Processing 3<sup>rd</sup>*, Pearson Prentice Hall, USA.
- [5] Putra, Dharma, (2010), *Pengolahan Citra Digital*, C.V. Andi Offset, Jakarta.
- [6] Kulkarni, A. D. (2001), *Computer Vision and Fuzzy-Neural Systems*, Prentice Hall, New Jersey.
- [7] Candra, Devy dkk. (2011), *Studi Pendeteksi Wajah dengan Metode Viola Jones*, Universitas Bina Nusantara, Jakarta.



*halaman ini sengaja dikosongkan*

## LAMPIRAN

### Lampiran 1. Spesifikasi Kamera Usb Emaging Source DF 31BU03

#### DFK 31BU03 Specification

##### GENERAL BEHAVIOR

Video formats & frame rate	1024 x 768 @ 30 FPS, UYVY @ 15, 7.5, 3.75 FPS 1024 x 768 @ 30 FPS, BYB @ 30, 15, 7.5, 3.75 FPS
Sensitivity	235 lx
Dynamic range	8 bit

##### INTERFACE (OPTICAL)

Bit (in Fiber)	yes
Sensor specification	Sony ICX204AK
Shutter	Global
Format	1/4"
Resolution	H: 1024 pixel, V: 768 pixel
Pixel size	H: 3.65 $\mu$ m, V: 4.65 $\mu$ m
Lens mount	C/CS

##### INTERFACE (ELECTRICAL)

Interface	USB
Supply voltage	4.5 VDC to 5.5 VDC
Current consumption	approx 300 mA at 5 VDC

##### INTERFACE (MECHANICAL)

Dimensions	H: 58.6 mm, W: 50.6 mm, L: 56 mm
Mass	260 g

##### ADJUSTMENTS (MANUAL)

Shutter	1 frame/s to 30 s
Gain	0 dB to 36 dB
Offset	0 to 511
Saturation	0% to 200%
White balance	-2 dB to +6 dB

##### ADJUSTMENTS (AUTOMATIC)

Shutter	1 frame/s to 30 s
Gain	0 dB to 36 dB
Offset	0 to 511
White balance	-2 dB to +6 dB

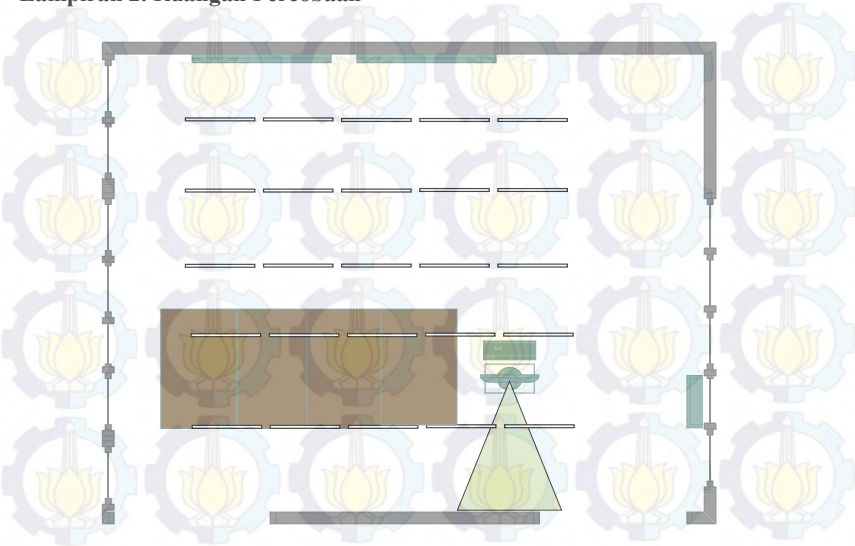
##### ENVIRONMENTAL

Temperature (operation)	-5 °C to 45 °C
Temperature (storage)	-25 °C to 60 °C
Humidity (operation)	20 % to 80 % non-condensing
Humidity (storage)	20 % to 95 % non-condensing

Subject change



Lampiran 2. Ruangn Percobaan

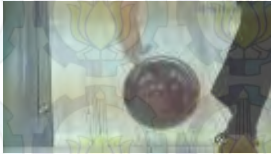


Gambar Lampiran 2. Ruangn Uji Pencarian Jarak Terjauh Pendeteksian

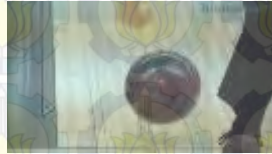


Gambar Lampiran 2b. Ruangn Uji Pendeteksian Objek

### Lampiran 3. Frame Pencarian Variabel Uji Coba 1



Frame ke-1



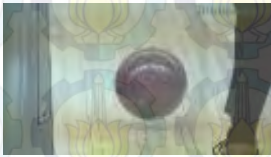
Frame ke-2



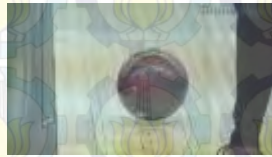
Frame ke-3



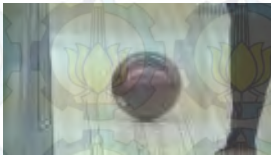
Frame ke-4



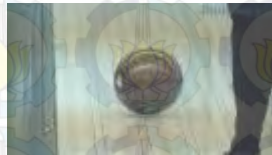
Frame ke-5



Frame ke-6



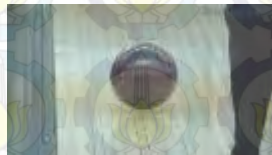
Frame ke-7



Frame ke-8



Frame ke-9



Frame ke-10

#### Lampiran 4. Analisa Kecepatan Satu Lintasan Objek

Tabel Lampiran 4a. Pendeteksian Real-Time Objek Merah Kondisi 1

No.	$\theta_o$	Frame ke-i	$\theta_i$	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1	5	1	8	1	0	0	0	#NUM!
2		2	3	1	0	0	0	0,091311
3		3	7	1	0	0	0	#NUM!
4	10	1	11	1	0	0	0	#NUM!
5		2	5	1	0	0	0	0,196977
6		3	5	1	0	0	0	0,196977
7	15	1	14	1	0	0	0	0,118854
8		2	13	1	0	0	0	0,165917
9		3	13	1	0	0	0	0,165917
10	20	1	19	1	0	0	0	0,133728
11		2	10	1	0	0	0	0,387596
12		3	14	1	0	0	0	0,314524
13	25	1	26	1	0	0	0	#NUM!
14		2	1	1	0	0	0	#NUM!
15		3	24	1	0	0	0	0,144015
16	30	1	30	1	0	0	0	0
17		2	16	1	0	0	0	0,54968
18		3	25	1	0	0	0	0,337053
19	35	1	30	0	1	0	0	0,347424
20		2	22	0	1	0	0	0,564687
21		3	29	1	0	0	0	0,381684
22	40	1	41	1	0	0	0	#NUM!
23		2	29	0	1	0	0	0,534013

24		3	18	0	1	0	0	0,758008
25	45	1	48	1	0	0	0	0,870413
26		2	24	0	1	0	0	0,769119
27		3	33	0	1	0	0	0,563673
28	50	1	50	1	0	0	0	0
29		2	25	0	1	0	0	0,862085
30		3	34	1	0	0	0	0,662965
31	55	1	55	1	0	0	0	0
32		2	28	0	1	0	0	0,910003
33		3	37	0	1	0	0	0,702044
34	60	1	59	1	0	0	0	0,117031
35		2	27	0	1	0	0	1,032382
36		3	34	0	1	0	0	0,88118

Tabel Lampiran 4b. Pendeteksian Real-Time Objek Merah Kondisi 2

No.	0 <sub>o</sub>	Frame ke-i	0 <sub>i</sub>	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1	5	1	8	1	0	0	0	#NUM!
2		2	6	1	0	0	0	0,091311
3		3	0	1	0	0	0	0,114315
4	10	1	12	1	0	0	0	#NUM!
5		2	6	1	0	0	0	0,217549
6		3	12	1	0	0	0	#NUM!
7	15	1	17	1	0	0	0	#NUM!
8		2	9	1	0	0	0	0,26999
9		3	15	1	0	0	0	0
10	20	1	19	1	0	0	0	0,133728
11		2	10	1	0	0	0	0,387596



12		3	16	1	0	0	0	0,261592
13		1	26	1	0	0	0	#NUM!
14	25	2	17	1	0	0	0	0,39622
15		3	22	1	0	0	0	0,248232
16		1	31	1	0	0	0	#NUM!
17	30	2	17	1	0	0	0	0,532424
18		3	10	1	0	0	0	0,628919
19		1	32	1	0	0	0	0,267121
20	35	2	17	1	0	0	0	0,656244
21		3	30	1	0	0	0	0,347424
22		1	40	1	0	0	0	0
23	40	2	11	1	0	0	0	0,844543
24		3	34	1	0	0	0	0,385557
25		1	45	1	0	0	0	0
26	45	2	29	1	0	0	0	0,663302
27		3	24	1	0	0	0	0,769119
28		1	50	1	0	0	0	0
29	50	2	29	0	1	0	0	0,780321
30		3	31	1	0	0	0	0,735403
31		1	58	0	1	0	0	#NUM!
32	55	2	17	1	0	0	0	1,096249
33		3	47	1	0	0	0	0,416112
34		1	59	0	1	0	0	0,117031
35	60	2	23	0	1	0	0	1,10606
36		3	51	1	0	0	0	0,419346

Tabel Lampiran 4c. Pendeteksian Real-Time Objek Hijau Kondisi 1

No.	$\theta_0$	Frame ke-i	$\theta_i$	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1	5	1	5	1	0	0	0	0
2		2	3	1	0	0	0	0,091311
3		3	5	1	0	0	0	0
4	10	1	9	1	0	0	0	0,098233
5		2	2	1	0	0	0	0,223628
6		3	8	1	0	0	0	0,135599
7	15	1	15	1	0	0	0	0
8		2	5	1	0	0	0	0,321152
9		3	16	1	0	0	0	#NUM!
10	20	1	19	1	0	0	0	0,133728
11		2	4	1	0	0	0	0,444675
12		3	14	1	0	0	0	0,314524
13	25	1	26	1	0	0	0	#NUM!
14		2	15	0	1	0	0	0,437019
15		3	18	1	0	0	0	0,372791
16	30	1	30	1	0	0	0	0
17		2	27	1	0	0	0	0,260948
18		3	3	1	0	0	0	0,67383
19	35	1	35	1	1	0	0	0
20		2	29	0	0	0	0	0,381684
21		3	9	0	0	0	0	0,751336
22	40	1	39	1	0	0	0	0,151723
23		2	17	0	1	0	0	0,772926
24		3	36	1	0	0	0	0,310755
25	45	1	47	1	0	0	0	#NUM!

26		2	30	0	1	0	0	0,639715
27		3	26	0	1	0	0	0,729165
28		1	51	1	0	0	0	#NUM!
29	50	2	39	0	1	0	0	0,527839
30		3	21	0	1	0	0	0,932849
31		1	57	1	0	0	0	#NUM!
32	55	2	49	0	1	0	0	0,349135
33		3	31	0	1	0	0	0,845823
34		1	64	1	0	0	0	#NUM!
35	60	2	38	0	1	0	0	0,783616
36		3	33	0	1	0	0	0,902756

Tabel Lampiran 4d. Pendeteksian Real-Time Objek Hijau Kondisi 2

No.	$\theta_o$	Frame ke-i	$\theta_i$	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1		1	8	1	0	0	0	#NUM!
2	5	2	5	1	0	0	0	0
3		3	0	1	0	0	0	0,114315
4		1	12	1	0	0	0	#NUM!
5	10	2	4	1	0	0	0	0,208777
6		3	11	1	0	0	0	#NUM!
7		1	19	1	0	0	0	#NUM!
8	15	2	5	1	0	0	0	0,321152
9		3	10	1	0	0	0	0,250751
10		1	22	1	0	0	0	#NUM!
11	20	2	13	1	0	0	0	0,336216
12		3	18	1	0	0	0	0,434056
13	25	1	28	1	0	0	0	#NUM!

14		2	18	1	0	0	0	0,549465
15		3	23	1	0	0	0	0,203232
16		1	32	1	0	0	0	#NUM!
17	30	2	11	1	0	0	0	0,618439
18		3	26	1	0	0	0	0,301481
19		1	38	1	0	0	0	#NUM!
20	35	2	0	0	1	0	0	0,787997
21		3	33	1	0	0	0	0,217108
22		1	38	1	0	0	0	0,216409
23	40	2	19	0	1	0	0	0,74223
24		3	20	1	0	0	0	0,725585
25		1	42	1	0	0	0	0,261409
26	45	2	28	1	0	0	0	0,686061
27		3	26	1	0	0	0	0,729165
28		1	46	1	0	0	0	0,293158
29	50	2	24	0	1	0	0	0,880823
30		3	37	1	0	0	0	0,584207
31		1	55	1	0	0	0	0
32	55	2	14	0	1	0	0	1,132435
33		3	46	1	0	0	0	0,447898
34		1	60	1	0	0	0	0
35	60	2	13	0	1	0	0	1,243513
36		3	46	1	0	0	0	0,567904



Tabel Lampiran 4e. Pendeteksian Real-Time Objek Ungu Kondisi 1

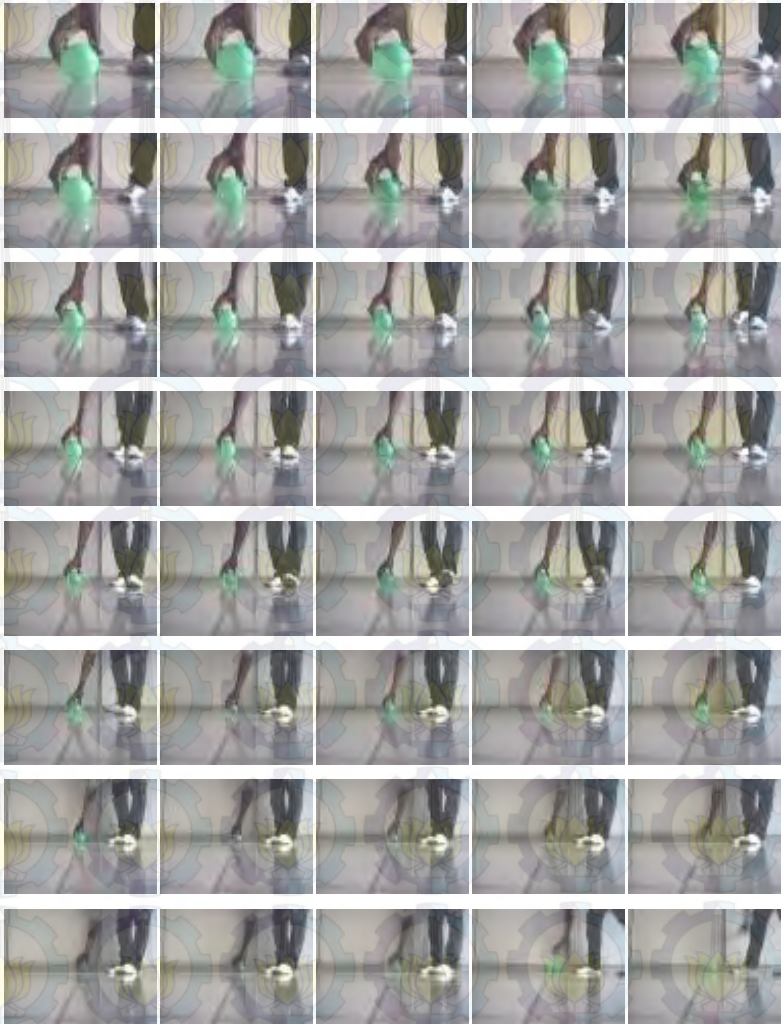
No.	$\theta_o$	Frame ke-i	$\theta_i$	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1	5	1	6	1	0	0	0	#NUM!
2		2	2	1	0	0	0	0,10469
3		3	7	1	0	0	0	#NUM!
4	10	1	10	1	0	0	0	0
5		2	3	1	0	0	0	0,217549
6		3	7	1	0	0	0	0,161793
7	15	1	15	1	0	0	0	0
8		2	8	1	0	0	0	0,286291
9		3	11	1	0	0	0	0,227925
10	20	1	19	1	0	0	0	0,133728
11		2	11	1	0	0	0	0,37248
12		3	14	1	0	0	0	0,314524
13	25	1	24	1	0	0	0	0,144015
14		2	11	0	1	0	0	0,499194
15		3	15	1	0	0	0	0,437019
16	30	1	28	0	1	0	0	0,212828
17		2	4	1	0	0	0	0,670403
18		3	27	1	0	0	0	0,260948
19	35	1	36	1	0	0	0	#NUM!
20		2	11	0	1	0	0	0,733175
21		3	35	1	0	0	0	0
22	40	1	42	1	0	0	0	#NUM!
23		2	4	0	1	0	0	0,889412
24		3	29	0	1	0	0	0,534013
25	45	1	45	0	1	0	0	0

26		2	18	0	1	0	0	0,870411
27		3	38	1	0	0	0	0,415322
28	50	1	50	0	1	0	0	0
29		2	38	1	0	0	0	0,978451
30		3	13	0	1	0	0	1,039652
31	55	1	57	0	1	0	0	#NUM!
32		2	38	0	1	0	0	0,676155
33		3	27	0	1	0	0	0,930192
34	60	1	62	0	1	0	0	#NUM!
35		2	53	1	0	0	0	0,355825
36		3	13	0	1	0	0	1,243513

Tabel Lampiran 4f. Pendeteksian Real-Time Objek Ungu Kondisi 2

No.	$\theta_0$	Frame ke-i	$\theta_i$	Objek Terdeteksi	Objek Tidak Terdeteksi			Kecepatan (m/s)
					Tipe 1	Tipe 2	Tipe 3	
1	5	1	6	1	0	0	0	#NUM!
2		2	0	1	0	0	0	0,114315
3		3	4	1	0	0	0	0,068418
4	10	1	14	1	0	0	0	#NUM!
5		2	13	1	0	0	0	#NUM!
6		3	1	1	0	0	0	0,227214
7	15	1	18	1	0	0	0	#NUM!
8		2	4	1	0	0	0	0,328791
9		3	14	1	0	0	0	0,227925
10	20	1	19	1	0	0	0	0,133728
11		2	19	1	0	0	0	0,133728
12		3	18	1	0	0	0	0,187862
13	25	1	31	1	0	0	0	#NUM!

14		2	11	1	0	0	0	0,499194
15		3	25	1	0	0	0	0
16		1	33	1	0	0	0	#NUM!
17	30	2	25	1	0	0	0	0,337053
18		3	14	1	0	0	0	0,618439
19		1	38	1	0	0	0	#NUM!
20	35	2	18	1	0	0	0	0,640034
21		3	25	1	0	0	0	0,495781
22		1	43	1	0	0	0	#NUM!
23	40	2	31	0	1	0	0	0,479455
24		3	27	1	0	0	0	0,583612
25		1	48	1	0	0	0	#NUM!
26	45	2	5	0	1	0	0	0,870411
27		3	42	1	0	0	0	0,415322
28		1	48	0	1	0	0	0
29	50	2	5	0	1	0	0	1,097361
30		3	42	1	0	0	0	0,556439
31		1	58	0	1	0	0	#NUM!
32	55	2	26	1	0	0	0	0,949763
33		3	38	1	0	0	0	0,676155
34		1	60	1	0	0	0	0
35	60	2	8	1	0	0	0	1,284806
36		3	51	1	0	0	0	0,418614





## Lampiran 5b. Hasil Pendeteksian Jarak Terjauh

frame ke: 1016  
bola x=319.000000 y=249.000000 r=81.467789  
frame ke: 1017  
bola x=315.000000 y=243.000000 r=77.201027  
frame ke: 1018  
bola x=319.000000 y=243.000000 r=78.447433  
frame ke: 1019  
bola x=315.000000 y=249.000000 r=74.706093  
frame ke: 1020  
bola x=303.000000 y=241.000000 r=70.604530  
frame ke: 1021  
bola x=299.000000 y=245.000000 r=67.911705  
frame ke: 1022  
bola x=291.000000 y=251.000000 r=53.814495  
frame ke: 1023  
bola x=293.000000 y=243.000000 r=51.623638  
frame ke: 1024  
bola x=295.000000 y=247.000000 r=49.244289  
frame ke: 1025  
bola x=291.000000 y=245.000000 r=44.384678  
bola x=281.000000 y=329.000000 r=32.449955  
frame ke: 1026  
bola x=287.000000 y=245.000000 r=42.755112  
frame ke: 1027  
bola x=281.000000 y=243.000000 r=38.078861  
frame ke: 1028  
bola x=279.000000 y=243.000000 r=36.345558  
frame ke: 1029  
frame ke: 1030  
frame ke: 1031  
bola x=291.000000 y=243.000000 r=32.015617  
frame ke: 1032  
bola x=283.000000 y=243.000000 r=28.178005  
frame ke: 1033  
bola x=291.000000 y=243.000000 r=29.529648

frame ke: 1034  
bola x=293.000000 y=243.000000 r=29.000000  
frame ke: 1035  
bola x=293.000000 y=243.000000 r=26.907248  
frame ke: 1036  
bola x=293.000000 y=243.000000 r=26.248808  
frame ke: 1037  
bola x=291.000000 y=245.000000 r=26.419689  
frame ke: 1038  
bola x=295.000000 y=245.000000 r=25.019991  
frame ke: 1039  
bola x=297.000000 y=245.000000 r=25.238857  
frame ke: 1040  
bola x=297.000000 y=245.000000 r=23.345232  
frame ke: 1041  
bola x=301.000000 y=245.000000 r=23.086790  
frame ke: 1042  
frame ke: 1043  
bola x=307.000000 y=243.000000 r=20.248455  
frame ke: 1044  
frame ke: 1045  
bola x=311.000000 y=245.000000 r=20.808651  
frame ke: 1046  
bola x=315.000000 y=245.000000 r=20.000000  
frame ke: 1047  
frame ke: 1048  
frame ke: 1049  
frame ke: 1050  
frame ke: 1051  
frame ke: 1052  
frame ke: 1053  
\*\*\*\*\*frame ke: 1054\*\*\*\*\*  
**bola x=345.000000 y=247.000000 r=19.416487**  
frame ke: 1055

## Lampiran 6. Kode Pemrograman Pendeteksian *Real-Time*

### Lampiran 6a. Kode Proyek Objek.h

```
#pragma once
#include <cv.h>
#include <highgui.h>
#include <string>

using namespace cv ;
using namespace std ;

class Object
{
public :
    Object() ;
    ~Object (void);

    Object(string name) ;

    int get_Posisi_X() ;
    void set_Posisi_X (int x) ;
    int get_Posisi_Y() ;
    void set_Posisi_Y (int y) ;
    int get_Posisi_R() ;
    void set_Posisi_R (int r) ;

    Scalar getThresmin() ;
    Scalar getThresmax() ;

    string getType()
    {
        return type ;
    }
    void setType(string t)
    {
        type = t ;
    }
}
```

```
void setThresmin (Scalar min) ;  
void setThresmax (Scalar max) ;
```

```
Scalar getColor()  
{  
    return Color;  
}
```

```
void setColor (Scalar c)  
{  
    Color = c ;  
}
```

```
private :  
int Posisi_X,  
    Posisi_Y,  
    Posisi_R;  
string type ;  
Scalar Thresmin,  
        Thresmax ;  
Scalar Color ;  
};
```

## Lampiran 6b. Kode Projek Objek.cpp

```
#include "Object.h"
```

```
Object :: Object()
```

```
{  
    setType ("Object") ;  
    setColor(Scalar (0, 0, 0));  
}
```

```
Object :: Object(string name)
```

```
{  
    setType (name) ;
```

```
    if (name == "ungu")
```

```
{  
        setThresmin (Scalar (50, 0 , 50  
    ));  
        setThresmax (Scalar (150, 73 , 125  
    ));  
        setColor (Scalar (219, 85  
    , 219 )) ;  
}
```

```
    if (name == "hijau")
```

```
{  
        setThresmin (Scalar (58, 90 , 40  
    ));  
        setThresmax (Scalar (183, 255 , 91  
    ));  
        setColor (Scalar (0 , 255  
    , 0 )) ;  
}
```

```
    if (name == "merah")
```

```
{  
        setThresmin (Scalar (0 , 0 , 154  
    ));
```



```

        setThresmax (Scalar (96, 79 , 255
    ));
    setColor (Scalar (0 , 0
, 255 )) ;
}

Object :: ~Object (void)
{}

int Object :: get_Posisi_X ()
{
    return Object :: Posisi_X ;
}
void Object :: set_Posisi_X (int x)
{
    Object :: Posisi_X = x ;
}

int Object :: get_Posisi_Y ()
{
    return Object :: Posisi_Y ;
}
void Object :: set_Posisi_Y (int y)
{
    Object :: Posisi_Y = y ;
}

int Object :: get_Posisi_R ()
{
    return Object :: Posisi_R ;
}
void Object :: set_Posisi_R (int r)
{
    Object :: Posisi_R = r ;
}

Scalar Object :: getThresmin()
{
    return Object :: Thresmin ;
}

Scalar Object :: getThresmax()
{
    return Object :: Thresmax ;
}

void Object :: setThresmin (Scalar min)
{
    Object :: Thresmin = min ;
}

void Object :: setThresmax (Scalar max)
{
    Object :: Thresmax = max ;
}

```

#### Lampiran 6c. Kode Projek Main

```
/*
----- Program Pendeteksian Untuk Mencari Jarak Terjauh
Pendeteksian -----
----- Tanggal      : 25-Juni-2014
*/

#include "cv.h"
#include "opencv2/core/core.hpp"
#include "opencv2/highgui.hpp"
#include "opencv2/imgproc.hpp"
#include "opencv2/opencv.hpp"
#include <sstream>
#include <string>
#include <vector>
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include <highgui.h>
#include <fstream>
#include <stack>
#include "stdio.h"
#include "conio.h"

#include "Object.h"

using namespace cv;
using namespace std;

string intToString(int number)
{
    std::stringstream ss;
    ss << number;
    return ss.str();
}

void morphOps(Mat &thresh)
{
    Mat erodeElement = getStructuringElement
```

```

        ( MORPH_RECT,
          Size(2,2));

    Mat dilateElement = getStructuringElement
        ( MORPH_RECT,
          Size(3,3));

    erode(thresh,thresh,erodeElement);
    erode(thresh,thresh,erodeElement);

    dilate(thresh,thresh,dilateElement);
    dilate(thresh,thresh,dilateElement);
}

void Tampilan_Pendeteksian
(
    vector <Object> theObjects,
    Mat &frame,
    Mat &Temp,
    vector<Vec3f>circles
)
{
    for ( int i = 0; i < theObjects.size(); i++)
    {
        rectangle
            (frame,

            Point(cvRound(theObjects.at(i).get_Posisi_X() -
theObjects.at(i).get_Posisi_R()),

            cvRound(theObjects.at(i).get_Posisi_Y() -
theObjects.at(i).get_Posisi_R())),

            Point(cvRound(theObjects.at(i).get_Posisi_X() +
theObjects.at(i).get_Posisi_R()),

            cvRound(theObjects.at(i).get_Posisi_Y() +
theObjects.at(i).get_Posisi_R()))),

```

```

        theObjects.at(i).getColor(),
        2,
        8);

    putText
        (frame,

        intToString(theObjects.at(i).get_Posisi_X())+ " ,
" + intToString(theObjects.at(i).get_Posisi_Y()),

        Point(theObjects.at(i).get_Posisi_X(),
                theObjects.at(i).get_Posisi_Y()),
        1,
        1,
        theObjects.at(i).getColor());

    putText
        (frame,
        theObjects.at(i).getType(),

        Point(theObjects.at(i).get_Posisi_X(),
                theObjects.at(i).get_Posisi_Y()-
                ((circles[i][2])+4)),
        1,
        2,
        theObjects.at(i).getColor());

    line
        (frame,

        Point(cvRound(theObjects.at(i).get_Posisi_X()-
        (0.25*theObjects.at(i).get_Posisi_R())), cvRound
        (theObjects.at(i).get_Posisi_Y())),

        Point(cvRound(theObjects.at(i).get_Posisi_X()+(0.
        25*theObjects.at(i).get_Posisi_R())), cvRound
        (theObjects.at(i).get_Posisi_Y())),
        theObjects.at(i).getColor());

```



```

        line
            (frame,
             Point(cvRound(theObjects.at(i).get_Posisi_X()),
                 cvRound
                 (theObjects.at(i).get_Posisi_Y()+(0.25*theObjects.at(i)
                 .get_Posisi_R()))),
             Point(cvRound(theObjects.at(i).get_Posisi_X()),
                 cvRound (theObjects.at(i).get_Posisi_Y()-
                 (0.25*theObjects.at(i).get_Posisi_R()))),
             theObjects.at(i).getColor());
    }
}

```

```

void Pendeteksian_Lingkaran
(Object theObject,
Mat image_threshold,
Mat image_gray,
Mat &Image_Camera)
{
    int Invers_Rasio_Resolusi = 2 ;
    int Jarak_Minimum_Lingkaran = 50 ;
    int Parameter_1 = 255 ;
    int Parameter_2 = 60 ;
    int Radius_Minimum = 5 ;
    int Radius_Maksimum = 200 ;

    vector <Object> objects;
    Mat Temp;
    image_threshold.copyTo(Temp);
    vector <Vec3f> circles;

```

HoughCircles

```
(Temp,  
circles,  
HOUGH_GRADIENT,  
Invers_Rasio_Resolusi,  
Jarak_Minimum_Lingkaran,  
Parameter_1,  
Parameter_2,  
Radius_Minimum,  
Radius_Maksimum);
```

```
bool objectFound = false;
```

```
for (int i = 0; i < circles.size(); i++)  
{
```

```
    cv::Point center(cvRound(circles[i][0]),  
                     cvRound(circles[i][1]));  
    int radius = cvRound(circles[i][2]);  
    printf("bola x=%f y=%f r=%f\n\r",  
           circles[i][0],
```

```
           circles[i][1],
```

```
           circles[i][2]);
```

```
    Object object;
```

```
    object.set_Posisi_X( circles[i][0]);  
    object.set_Posisi_Y( circles[i][1]);  
    object.set_Posisi_R( circles[i][2]);  
    object.setType(theObject.getType());  
    object.setColor(theObject.getColor());
```

```
    objects.push_back(object);
```

```
    objectFound = true;
```

```
    if(objectFound ==true)
```

```

    {
        Tampilan_Pendeteksian
        (objects,
        Image_Camera,
        Temp,
        circles);
    }
}

```

```

int main(int argc, char** argv)
{

```

```

    Mat Image_Camera,
        image_threshold
        ,
        image_threshold_ungu,
        image_threshold_hijau,
        image_threshold_merah,
        image_gray
        ,
        image,
        frame;

```

```

    VideoCapture capture(1);
    CvVideoWriter *video =
        cvCreateVideoWriter("video/video.AVI",
        CV_FOURCC('M', 'J', 'P', 'G'), 3, Size(640, 480));
    capture.set(CV_CAP_PROP_FPS,30);

```

```

    waitKey (2);

```

```

while (true)
{

```

```

    freopen("myfile.txt", "a", stdout);
    static unsigned int counter = 0;
    char FileNameOut[]
    = "imgsout/img0000.jpg" ;
    char FileNameOutOriginal[] =
    "Original/img0000.jpg";

```

```

        int LenExtension = 4;

        int FileNameLengthOut
        = strlen(FileNameOut)
        ;
        int FileNameLengthOriginalOut
        =
        strlen(FileNameOutOriginal)
        ;
        int index1, index2;

        printf
        (
            "\n*****frame ke: %d
            *****\n",
            counter++);

        int NoDigits
            = ceil(log((double)(counter)) /
            log(double(10)));
        char *TempNumber
            = (char
            *)malloc(sizeof(char)*NoDigits);
        sprintf(TempNumber, "%d", counter);

        for (index1 = 0; index1 < NoDigits;
        index1++)
        {
            FileNameOut[FileNameLengthOut -
            LenExtension
            - NoDigits + index1] =
            TempNumber[index1];

            FileNameOutOriginal[FileNameLengthOriginalOut
            - LenExtension - NoDigits +
            index1]
            = TempNumber[index1];
        }
        capture >> Image_Camera
        ;

```



```

        resize(Image_Camera, Image_Camera,
Size(640, 480), 0, 0, 2);
        image = Image_Camera.clone();
        frame = Image_Camera.clone();

        if (!capture.isOpened())
        {
            printf("KAMERA USB TIDAK
TERDETEKSI\n");
            return -1;
        }

        Object ungu ("ungu"), hijau
("hijau"), merah ("merah");

        cvtColor (frame, image_gray,
COLOR_BGR2GRAY);
        threshold (image_gray,
image_threshold, 150, 255, 0);

        cvtColor
(Image_Camera,
image_gray,
COLOR_BGR2GRAY);
        inRange
(Image_Camera,
hijau.getThresmin(),
hijau.getThresmax(),
image_threshold);
        morphOps (image_threshold);
        image_threshold_hijau=image_threshold.clone();

        GaussianBlur(image_threshold,image_threshold,
Size(9, 9), 2, 2);
        Pendeteksian_Lingkaran(hijau,
image_threshold, image_gray, Image_Camera);

```

```
        cvtColor
        (Image_Camera,
         image_gray,
         COLOR_BGR2GRAY);

        inRange
        (Image_Camera,
         ungu.getThresmin(),
         ungu.getThresmax(),
         image_threshold);
        morphOps (image_threshold);

        image_threshold_ungu=image_threshold.clone();

        GaussianBlur(image_threshold,image_threshold,
        Size(9, 9), 2, 2);
        Pendeteksian_Lingkaran(ungu,
        image_threshold, image_gray, Image_Camera);

        cvtColor
        (Image_Camera,
         image_gray,
         COLOR_BGR2GRAY);

        inRange
        (Image_Camera,
         merah.getThresmin(),
         merah.getThresmax(),
         image_threshold);
        morphOps(image_threshold);

        image_threshold_merah=image_threshold.clone();

        GaussianBlur(image_threshold,image_threshold,
        Size(9, 9), 2, 2);
        Pendeteksian_Lingkaran(merah,
        image_threshold, image_gray, Image_Camera);
```

```
        IplImage* Image_Ipl =  
        cvCloneImage(&(IplImage)Image_Camera);  
  
        fclose(stdout);  
        imshow ("Image_Camera"  
        ,Image_Camera )      ;  
        imwrite      (FileNameOut  
        ,Image_Camera      )      ;  
        cvWriteFrame(video, Image_Ipl);  
        imwrite      (FileNameOutOriginal,frame  
        )      ;  
        waitKey(10)      ;  
    }  
    return 0;  
}
```

## BIODATA PENULIS



Penulis, Muhamad Al Imron dilahirkan di kota Tulungagung pada tanggal 17 Oktober 1991. Putra ketiga dari tiga bersaudara pasangan Bapak Djaenuri dan Ibu Komsiyah ini memulai pendidikannya di SDN Tugu 3 (1997 – 2003). Dilanjutkan MTs Negeri 1 Karangrejo (2003 – 2006), dan SMA Negeri 1 Karangrejo (2006 – 2009). Kemudian pada tahun 2007 penulis melanjutkan pendidikan ke tingkat perguruan tinggi di Institut Teknologi Sepuluh Nopember (ITS) Surabaya jurusan S1 Teknik Mesin melalui jalur SNMPTN (Seleksi Nasional Masuk Perguruan Tinggi Negeri).

Di jurusan Teknik Mesin, penulis mengambil bidang Manufaktur laboratorium Perancangan dan Pengembangan Produk dan memiliki ketertarikan di bidang image processing. Selama di bangku perkuliahan selain belajar, penulis mengikuti kegiatan organisasi yaitu LDJ. Ash Shaff di Teknik Mesin. Penulis dapat dihubungi melalui alamat email [cetoel09@gmail.com](mailto:cetoel09@gmail.com).